

國立高雄第一科技大學
資訊管理系
碩士論文

應用活動理論與本體論建構一高階主管
知識系統

研究生：李湘茹

指導教授：許蒞彥

中華民國九十六年六月

應用活動理論與本體論建構—高階主管知識系統

研究生：李湘茹 Shang-Ju Lee

指導教授：許蒞彥 Li-Yen Shue

國立高雄第一科技大學

資訊管理系

碩士論文

A Thesis Submitted to
Department of Information Management
National Kaohsiung First University of Science and Technology

In Partial fulfillment of the Requirements

For the Degree of Master

In

Information Management

June 2007

Yenchao, Kaohsiung, Taiwan, Republic of China

中華民國九十六年六月

國立高雄第一科技大學學位論文考試審定書

資訊管理系(所) 碩士班

博士班

研究生 李湘茹 所提之論文

應用活動理論與本體論建構—高階主管知識系統

碩士

經本委員會審查，符合

學位論文標準。

博士

學位考試委員會

召集人 林珮君 簽章
委員 林珮君 陳慶文
許桂亭

指導教授 許桂亭 簽章

系主任 陳文生 簽章

中華民國 96 年 06 月 15 日

保存期限：永久

表單編號：ACAA-3-02-3703

應用活動理論與本體論建構一高階主管知識系統

摘要

現今企業裡，高階主管都必須揹負企業生存及發展的責任，這些責任包括追求利潤、提升競爭能力、永續經營。高階主管要達到以上這些責任，需要有人提供他們在企業營運相關的需求知識，這些知識分別為成本、行銷、文化、環境、競爭者、客戶、員工。然而目前現行的高階主管資訊系統，雖然能提供資訊給高階主管做即時存取，但是這些資訊都是由各部門自行彙整，並沒有將資訊進一步有效分析，變成深入且有效的知識來配合高階主管認知想法中所需的知識。因此本研究將運用活動理論(Activity Theory)從高階主管的管理營運的認知層次(Cognitive Level)加以分析，進而建立其思考的心智流程(Mental Process)及其運作所需的相關的知識需求及內容。在系統的建置上將運用本體論(Ontology)來整合企業基本功能應有的知識及知識相互之間的關係，並以 Protégé 及 JAVA 建構一高階主管知識系統。

關鍵字：本體論、活動理論、認知層次、心智流程、Protégé

Development of an Executive Information System with Activity Theory and Ontology

Student: Shang-Ju Lee

Advisor: Li-Yen Shue

Department of Information Management

National Kaohsiung First University of Science and Technology

Abstract

This research examines the feasibility of applying Activity Theory and Ontology to develop an executive knowledge system for top management. Present EIS was essentially developed as a hybrid version of tradition information systems and decision support systems, and is not capable of supplying in-depth knowledge that is needed by executives before proper decisions can be made. We apply Activity Theory to examine the cognitive level of knowledge conceptualizations that are in the minds of executives, and apply Ontology to explore the natures of various functionalities of the enterprise and relations between them. We propose six knowledge dimensions that are essential for executives as a result of mapping the two together. With the actual responsibilities and the corresponding knowledge needs of the executive of a chemical company, who is dissatisfied with what the present EIS can supply to him, we develop an executive knowledge system.

Keywords : Ontology 、 Activity Theory 、 Cognitive Level 、 Mental Model 、 Protégé

誌謝

首先，真的很感謝我的指導教授：許蒞彥老師，因為他的循循善誘，讓學生的論文得以順利完成。在這兩年之中，老師身兼圖書館館長常常是忙的不可開交，但是他卻每週都會撥出時間與學生開會討論，在他的諄諄教誨及悉心指導之下，學生總是深刻的感受到老師的細心引導，甚至在研究方法或思惟方面也總是不厭其煩的啟迪學生思考。老師那嚴謹的教學方式更是無人能及，面對學生有時的愚笨，也經常耐心的開導學生該如何去解決難題，學生真的感到受益匪淺、永銘於心。

在此更感謝我的口試委員：成功大學林珮琚老師及本校陳慶文老師，感謝他們撥空審閱我的論文，並在論文口試期間給予學生寶貴的意見，使得我的論文得以更加完善及嚴謹，真的很感謝你們。

在這碩士的兩年中，感謝與我一起學習的朋友們，千子、承益、長謀、智仁、家豪等，總是在我遇到難題時點醒我、幫助我，一直鼓勵及支持我，讓我更有勇氣及信心來完成我的論文，有時遇到困難與挫折，也會彼此互相扶持、互相討論，我相信這樣的情誼絕對是支撐我完成論文的動力。最後，還要感謝我的父母親及文崇，一直默默不斷的支持我，每當我遇到挫折，他們總是我最大的避風港，有了你們的鼓勵，讓我順利完成此論文，願與你們分享這成果與喜悅，特此感謝。

目錄

摘要.....	I
Abstract.....	II
誌謝.....	III
目錄.....	IV
圖目錄.....	VI
表目錄.....	VII
壹、緒論.....	1
一、研究背景.....	1
二、研究動機.....	2
三、研究目的.....	3
四、研究流程.....	4
五、論文架構.....	5
貳、文獻探討.....	6
一、高階主管資訊系統.....	6
二、活動理論.....	10
三、本體論.....	18
(一)本體論定義.....	18
(二)本體論語言.....	19
(三)本體論工程.....	20
四、Protégé.....	23
(一)Protégé 的特色.....	23
(二)Protégé 的知識模組.....	25

參、高階主管知識分析.....	27
一、公司知識內容分析.....	28
二、高階主管認知層次分析.....	33
三、高階主管知識內容分析.....	41
(一)語意網(Semantic Web).....	41
(二)類別的制定.....	45
肆、系統分析與設計.....	49
一、系統概念分析.....	49
二、系統設計.....	52
三、系統建置與功能說明.....	56
伍、結論與未來研究方向.....	62
一、結論.....	62
二、未來研究方向.....	64
陸、參考文獻.....	66
一、中文部份.....	66
二、英文部份.....	66

圖目錄

圖 1.4.1 研究流程.....	5
圖 2.2.1 活動階層性架構.....	12
圖 2.2.2 個人層級的活動關係圖.....	13
圖 2.2.3 活動系統.....	13
圖 2.3.1 本體論語言架構.....	20
圖 2.3.2 TOVE 本體論工程六階段.....	21
圖 2.4.1 Protégé 的系統架構.....	24
圖 3.2.1 活動理論與高階主管決策知識心智流程圖.....	33
圖 3.2.2 追求利潤階層性架構.....	38
圖 3.2.3 提升競爭能力階層性架構.....	39
圖 3.2.4 永續經營階層性架構.....	40
圖 3.3.1 Level 1 企業功能知識語意圖.....	42
圖 3.3.2 企業各功能知識層次語意圖.....	43
圖 3.3.3 企業功能知識內容完整語意圖.....	44
圖 3.3.4 OWL建置之領域知識本體論.....	46
圖 3.3.5 關係屬性.....	47
圖 3.3.6 知識連結關係本體論.....	48
圖 4.1.1 系統概念圖.....	50
圖 4.1.2 系統架構圖.....	53
圖 4.1.3 系統功能流程圖.....	55
圖 4.2.1 使用Protégé平台所建構的高階主管知識本體論.....	56
圖 4.2.2 高階主管知識系統首頁.....	58
圖 4.2.3 知識需求與OWL本體論的連結.....	59
圖 4.2.4 知識需求相關的知識內容圖.....	60

圖 4.2.5 知識類別分析圖.....61

表目錄

表 2.2.1 藉由資訊科技的觀點將外在執行層結合活動系統之分類表.....16

表 2.2.2 活動、行為、操作三個層級的舉例說明.....17

表 3.1.1 管理營運的認知層次所需的知識內容.....30

表 3.1.2 知識面與認知層次及知識內容相關表.....32



壹、緒論

一、研究背景

在全球化的商業經營模式下，企業的經營模式變的越來越複雜，各部門的分工越來越細，經營型態也越來越多樣化，讓企業在管理上更加的困難，再加上專業分工使得企業內部所要呈現的資訊越來越多且分散，無法集中。這些資訊大部份是用來提供高階主管企業營運的相關知識，但是這些知識只是各個部門自行將資訊彙整，並沒有辦法將這些資訊有效、深入的分析提供給高階主管使用。高階主管經營企業所面臨的工作環境屬於高複雜性，所要達到的工作責任也不只一個。當高階主管所需揹負的工作責任越多，背後所提供的知識也會跟著增加，然而高階主管為了達到工作責任，必須將所有的工作責任之間的關係環環相扣，也因如此，導致背後所需的知識層面亦會跟著變動，在如此循環的過程之下，高階主管的知識需求乃成為變動的性質。因此，所面臨的資訊就算即時、正確也不能有效的根據高階主管的需求來給予適當的資訊，進而影響高階主管決策的準確性。

高階主管決定企業營運未來的走向跟決策時，必須瞭解更多的資訊，然而各部門資訊越來越多且複雜，導致企業都偏向於導入高階主管資訊系統(Executive Information System, EIS)，因為 EIS 能提供高階主管存取內外部資訊，幫助高階主管制定決策。雖然 EIS 提供這麼多功能輔助高階主管，但是在呈現資訊方面卻沒有辦法滿足高階主管的需求，EIS 只能提供高階主管存取資訊，但是並不能得知什麼樣的資訊才是高階主管最需要的。

目前大多數企業都導入 EIS，導入的案例顯示出僅有 40%成功，其餘的 60%

都是失敗的 (Watson et al., 1991) ，且失敗的原因都在於無法滿足高階主管的需求，本研究回顧 EIS 文獻發現以下兩點的問題：(1) 只有極少數學者是偏向於 EIS 開發實務的研究，因為要如何開發出一個與高階主管資訊需求相符合的 EIS 系統，是許多研究者的困難及挑戰 (Watson et al., 1991) ；(2) 發展有關 EIS 的研究幾乎都是在 1985~2000 年，這樣的數據顯示不禁懷疑是否因為過去導入 EIS 成功的案例不多，才會導致於 2000 年以後研究 EIS 的相關論文少之又少。

由於過去的 EIS 都無法滿足高階主管的需求，才會導致有將近 60% 的失敗案例。根據此數據的顯示，陸續有學者對於如何促使 EIS 成功的議題，探討的相當多。因此，本研究認為 EIS 對於企業是一門非常重要的課題，成功的 EIS 應為可以清楚瞭解高階主管的知識需求及適時的提供他想要的知識，並不是讓高階主管從複雜的知識裡找尋。

二、研究動機

高階主管所做的決策往往關係到企業未來的走向，但是決策的好壞，則導因於所得到的資訊。就企業來說，目前所面臨的問題，電子化的企業模式加快了作業的速度，到達了零時差的境界，但也帶來了爆炸的資訊流；在人工無法完全閱讀的情況下，使得企業需要藉助龐大的資料庫以收集資料和分析資訊，進而轉換成有用的資訊，以提供企業內外的資訊以便高階主管做決策。

企業如何確保資訊是否正確性、價值性、重要性，所依賴的也只有龐大的資料庫，但資料庫所提供的只有數據跟記錄，並沒有辦法提供有用的資訊，更加沒有辦法針對高階主管的需求做出適切的回應，因為高階主管的心智需求屬於變動性，傳統的 EIS 所提供的固定功能模式只能針對特定的問題，無法根據高階主管的變動需求給予適當的知識層面。

針對高階主管和企業在知識需求方面的問題做出了兩點的歸納(Rainer and Watson, 1995 ; Watson et al., 1991; Young and Watson, 1995)：

- (一) 高階主管：高階主管肩負整個企業的責任，所考慮的層面也較廣。在此所指的層面為高階主管企業營運所需要考量的知識面；也因決策上的特性與一般主管大不相同，因此瞭解高階主管內在認知及需求便成為相當重要。
- (二) EIS 功能有限：傳統 EIS 所提供的功能為固定的功能模式，無法因應高階主管的心智流程及需求的變動。系統設計者也不瞭解高階主管的企業營運知識，只站在自己的角度，覺得有高超的技術就可以幫助高階主管做決策，對於高階主管的需求並沒有辦法提供。

有鑑於此，本研究認為運用活動理論(Activity Theory)分析高階主管知識需求及透過本體論(Ontology)來得知企業營運的相關知識內容。

三、研究目的

由於 EIS 無法滿足高階主管的心智需求，是因為 EIS 所提供的功能為固定模式，只能依原本所給予的功能進行判斷，但高階主管的心智需求為變動性，傳統的功能模式不能因應需求的變動。因此，本研究應用活動理論的概念將高階主管的內心需求及認知層次架構予以分析，以求瞭解高階主管的心智流程，並結合本體論(Ontology)的概念塑模領域知識，此知識為高階主管經營企業所需的知識內容，最後運用本體論語言 OWL (Ontology Web Language, OWL)建構一領域知識本體論及運用 JAVA 建立一相互溝通的機制，以提昇知識庫內容的可再用性 (Reusability)、可理解性 (Understandability) 與可擴充性 (Extendibility) (Chandrasekaran et al., 1999; Sugumaran and Storey, 2002)。

本研究的子目標為：

- (一) 應用活動理論(Activity Theory)分析出高階主管所需的知識需求及認知層次架構，以確立高階主管知識需求內容及認知層次相互之間的關係，進而建立其高階主管的心智流程。
- (二) 經由分析得知的認知層次架構來導出企業功能知識，並應用本體論(Ontology)來分析企業功能知識之間的相互關係。
- (三) 將活動理論與本體論兩者結果以 OWL 本體論語言建立一個知識本體論，並運用 Java 建構一高階主管知識系統。

四、研究流程

本研究建構一高階主管知識系統，主要是從高階主管本身擷取知識，並對其知識進行分析，分析的部份可以分為兩個理論進行，一為活動理論、二為本體論，最後將分析完的知識透過適當的表示來加以呈現。本研究的研究架構主要分為五個階段：

- (一) 藉由訪談一家公司之高階主管實際經營情況，以分析一般高階主管的心智流程及知識需求為何。
- (二) 應用活動理論(Activity Theory)分析高階主管的心智需求，以建立高階主管的心智流程及運用學者 Leontiev (1981)所提出的活動階層性架構(Hierarchical Structure of Activity)來分析高階主管的認知層次架構。
- (三) 運用本體論(Ontology)，及多倫多大學所發展的本體論方法 TOVE (Toronto Virtual Enterprise, TOVE) (Uschold and King, 1995; Kim, 2002; Fernandez, 1999)來建立高階主管知識內容的領域本體論，其知識內容是包括企業營運相關知識。

- (四) 將活動理論分析完的高階主管心智流程及認知層次架構再結合本體論所分析的企業營運相關知識，透過史丹佛大學所發展的 Protégé 知識平台呈現。
- (五) 最後運用 Java 語法 建立相互溝通的使用者介面機制，以建構一完整之高階主管知識系統。

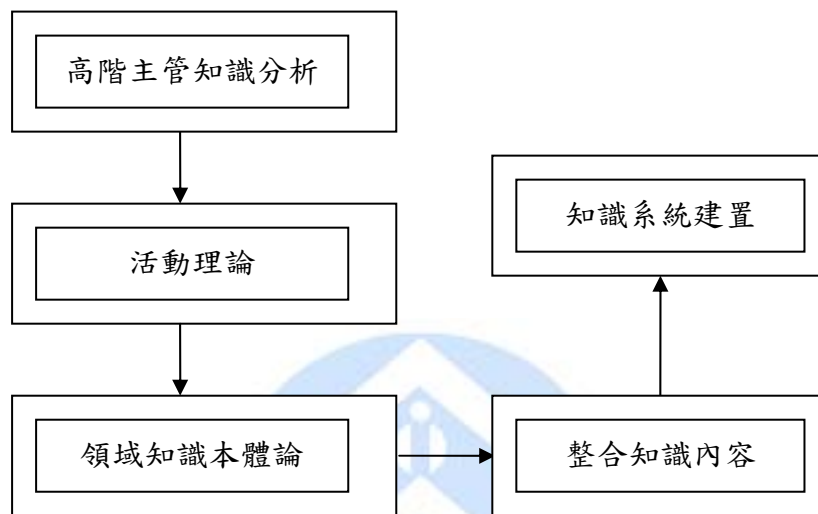


圖 1.4.1 研究流程

五、論文架構

本論文共分為五章，第一章緒論，描述本研究之研究背景、動機與目的，並說明整個研究的流程架構。第二章文獻探討，首先為高階主管資訊系統相關之內容，接著談到活動理論所要分析的兩個部份：一為高階主管的心智流程模式；二為高階主管的認知層次架構，再來談到本體論的理論與應用，最後則介紹 Protégé 的目的與功能等。第三章為研究方法，將詳細說明本研究所進行的各階段內容，包括活動理論與本體論分析。第四章則介紹本研究所建置的高階主管知識系統，描述該系統之架構、設計與功能。第五章為結論和未來研究方向討論。

貳、文獻探討

一、高階主管資訊系統

高階主管資訊系統(EIS)是繼決策支援系統(Decision Support Systems, DSS)之後所發展出來的觀念，DSS 一開始是設計被用來滿足高階主管的特定需求，可以提供資訊擷取、分析等等，但是高階主管的決策需求通常很廣泛且內容會一直擴充，這些特徵反應出高階主管的決策環境缺乏了結構性、具有高度不確定性及對於低階層級的需求的詳細性，因此漸漸衍伸出 EIS。EIS 崛起於 1980，此年代學者對於 EIS 並未深入的探討，只針對企業高階主管的經營策略、如何提升競爭優勢、影響組織的內外部因素做一個概念上的討論，此時 EIS 並無確切的定義。EIS 一詞最早是由 Rockart and Treacy (1982) 所提出，直到 1980 末期至 1990，EIS 才漸漸受到各方學者的注視及大力的探討研究。

EIS 由學者 Paller and Laska (1990) , Turban and Watson (1989) 定義為「一個電腦系統，且提供高階主管容易存取內部及外部的資訊，這些資訊是與企業相關的關鍵成功因素(Critical Success Factors, CSF)」。學者 Young and Watson (1995) 亦將 EIS 定義為「一個可開發的系統，且可以幫助高階主管存取資訊」。綜合以上兩位學者的觀點，本研究認為 EIS 為一資訊系統，能讓高階主管存取資訊及輔助高階主管做決策。

EIS 可以讓高階主管存取所需的內外部資訊，但是高階主管需要這些資訊主要是為了要實踐他們的工作責任(Aguilar, 1967; Daft and Lengel, 1986)，因為高階主管只有在真正實踐工作責任時，才能讓企業生存發展下去，因此以下將介紹高階主管所需的內外部資料(Walters et al., 2003)：

(一) 內部資訊

- 1、市場：客戶的資料庫、市場區隔、客戶追蹤。
- 2、產品研發與設計：新產品的特色、產品的專利權。
- 3、基本的工程師：只具備基本的知識搜集及管理能力。
- 4、財務管理：資產管理、現金、投資。
- 5、成本控制：控制各部門的成本開銷、降低一般的日常費用、成本節約。
- 6、作業效率：經濟規模、減少浪費、生產、不良率、生命週期。

(二) 外部資訊

- 1、市場環境：產業追蹤、客戶的喜好、產業促銷的改變。
- 2、技術環境：在產品及流程上有要新的突破、自動化、資訊科對企業造成的衝擊。
- 3、競爭環境：競爭者的相關活動、買方、供應商、新競爭者的進入、產業結構的改變。
- 4、政策/法律環境：法律上官方的行為、政府資金的來源/津貼、法律對企業的影響。
- 5、經濟環境：利率、通貨膨脹、存款率、貨幣變動、外匯率。
- 6、社會與文化環境：社會趨勢、價值的變化、道德規範、流行風尚、環境的保護。

EIS 除了提供內外部資訊之外，還必須有其他的功能來輔助高階主管，由於隨著時間的過去，EIS 發展到至今，所開發的功能特色也越來越多，以下將列點說明：

(Watson et al. 1991; Young and Watson 1995)

- (1) 針對單一個高階主管使用
- (2) 擷取、過濾、歸納且追蹤關鍵成功因素(Critical Success Factors, CSF)

- (3) 提供線上存取、趨勢分析、例外報告、和向下挖掘(所謂的向下挖掘指的是可以允許使用者去存取更詳細或是概括的資料)
- (4) 存取且整合更廣泛的內外部資料
- (5) 提供友善的使用者介面讓使用者可以在短時間內上手
- (6) 可直接由高階主管使用，不需要透過媒介
- (7) 可呈現圖、表、和文字資訊。
- (8) 可監督關鍵績效指標(Key Performance Indicators, KPI)

雖然 EIS 具有這麼多功能，導致於現今已有許多組織已經開發 EIS 系統，但是從最近的研究顯示出 EIS 系統是屬於高風險，且導入 EIS 的組織有將近 60% 皆為失敗(Watson et al., 1991)，更指出其失敗的原因有四點(Rainer and Watson, 1995 ; Watson et al., 1991; Young and Watson, 1995)：

- (1) 高階主管對 EIS 的抗拒：通常在於高階主管對於電腦本身的不適應、及打字技巧且高階主管也認為不必使用系統所提供的功能就能解決企業所面臨的問題。
- (2) 高階主管的工作性質：高階主管的工作為繁雜，行程忙碌。系統設計人員因為不瞭解高階主管的工作性質，所設計出的 EIS 就沒有辦法符合高階主管的需求。
- (3) EIS 的功能：系統設計者所提供的功能或技術對高階主管而言太過於困難，導致高階主管不擅長使用，再加上如果系統太過困難，就必須要有事前訓練，但是對於高階主管而言，這些事前訓練對他在經營企業應該瞭解的知識並無太直接關係。另外，傳統 EIS 的固定功能模式也無法因應高階主管知識需求的變動。
- (4) 無法提供有效的資訊需求：系統設計者對於高階主管應具有的企業營運知識不瞭解，只站在自己的角度，覺得只要有高超的技術就可以幫助高階主管做法

策，常常忽視高階主管的資訊需求，且因為高階主管必須根據企業營運所須的資訊需求時常更改決策，導致 EIS 無法即時更新，再加上 EIS 無法針對企業營運時所碰到的特定問題加以解決。

除了以上四點 EIS 導入失敗的原因之外，還有許多學者也對於 EIS 提出許多見解，例如 EIS 協會在 1988 及 1989 年提到 EIS 必須要能針對高階主管的需求去設定，也說出 EIS 要與高階主管一直在改變的需求做同步更新(Stecklow, 1989)。學者(McClatchy, 1990)也提到不只是 EIS 的問題，還有高階主管本身，他提到最主要的問題在於高階主管無法知道自己要的是什麼，也無法將自己要的東西詳細的描繪出來。甚至也有學者(Walters et al., 2003)提出應該要將 EIS 定位於一個可以在對的形式裡提供適當、適切的資訊給高階主管使用；另一位學者(Simon, 1957)也提到高階主管的認知裡有些許的限制，這些限制指的是高階主管所需的資訊必須經過篩選並列出優先的選擇，這句話的意思就是如果要有效的將資訊傳達給高階主管並幫助他們做出決策，EIS 則必須要知道哪一個資訊的類別對於高階主管是最重要且是他們最常用到的知識。

提到了這麼多學者對 EIS 的見解，可是到底該如何滿足高階主管的需求，及什麼是高階主管的工作性質，說明如下：

(1) 工作性質：

兩位學者(Watson and Frolick, 1993)提到高階主管的工作內容是非常多樣性，及概念性，且不連續性；且高階主管喜愛口頭上的溝通，非書面上的文字，因為他們認為這樣有助於資訊的交換；口頭上的溝通像是閒聊、思想觀念、見解評價、預測、說明解釋。此外，高階主管的工作大部份都是非結構化、非慣例、及大範圍的，因此大部份企業裡的高階主管都會有一個日常工作事項或是議程，

這個日常工作事項或是議程的內容皆是企業的目標、優先處理的事項、策略及規劃，而且都不是屬於文件可以描述的，還有高階主管也會去建立一個關係網絡，把組織內外部跟議程裡有相關的所有人員結合起來(Kotter, 1982)。

(2)滿足高階主管需求：

高階主管的資訊需求大部份都是從內外部資料的來源所取得的，為了要取得這些資源，高階主管總是在複雜的知識裡尋找，因為系統設計者無法有效提供即時且適當的資訊給高階主管，因此 EIS 開發人員必須要從不同角度去考慮使用者的心態、及每個高階主管的需求，只有站在高階主管的角度去考量，才能真正的為高階主管解決他們所面臨的問題，也才能為公司帶來獲利及生存發展的空間。

EIS 為了讓高階主管存取內外部資訊，就必須要滿足高階主管的需求及他的工作責任，但是從過去學者的研究也說明了 EIS 目前不足的缺點都是在於無法確切的得知高階主管心裡在想什麼及滿足高階主管的需求；因此本研究將運用活動理論將高階主管的心智流程予以分析，瞭解高階主管心裡的思維才能清楚他所要知識為何，也就能進一步滿足他的需求，並幫助高階主管做出適當的決策，解決企業營運時所面臨的問題。

二、活動理論

活動理論(Activity Theory)最早是起源於1920年~1930初期，由俄國人Vygotsky、Rubinshtein、Leontiev等學者所發展出來的，一開始是被用來探討人類電腦互動(Human Computer Interaction, HCI)。因此可以提供一個廣泛的概念性架構，此架構為描述HCI活動是如何開發，有什麼情境及結構。之後學者Vygotsky(1978)提出活動理論是藉由人們執行活動時可能會有一連串的行為所組成並達到

預期的成效。學者Engeström (1987)也對活動理論下定義，他認為活動理論是用來描述及分析活動的理論。之後諸位學者對於活動理論廣泛的定義則是變成活動理論為哲學及多準則方針的架構，這個架構是將人們每天所執行的活動及過程做一個研究探討，且可以同時的將個人層級與社會層級的活動與以連接。因此本研究認為活動理論可以將人類所採取的活動結合工具、環境、角色、規則、對象彼此相互影響進而達到預期的成效，且可將人們心裡所想的實際轉換成所採取的活動來實踐。

活動理論其實並不是一個方法或是理論架構，它最主要是提供方針、原則，這個方針在基本的概念性描述上，可用四個部份說明(Victor and Bonnie, 1997)：

(一) 活動階層性架構

活動理論分析的單位有三個階層，第一層為活動(Activity)到動機(Motive)，這裡所指的是有什麼樣的動機(Motive)及目的去執行活動，到第二層是行為(Actions)到目標(Goal)，代表有了想要執行的活動之後要採取什麼行動才能呼應此活動，並進而達到目標，第三層則是操作(Operations)到情境(Conditions)，如果要實踐此行為就必須要透過操作，操作最主要是要能根據不同或是目前的情境下所做的調整，因為活動理論就是要能根據持續變化的情況來採取接下來所要做的活動，因此每個層級都會相互呼應。

1. Activity是指一個人所要執行的活動(Activity)有什麼樣的目的及動機(Motive)，且活動是屬於長期所形成的目標(Goal)，所以不可能一次就能完成，要分成多個階段，因此需要短期的活動來分別執行使其達到目的；這裡指的短期活動是指一連串的行為及操作。
2. Actions如同一個人的行為(Actions)，且為了要達到所想要完成的目標時，會採

取很多行為來應變，因此在這裡的行為將會一直不斷的轉換來因應目前的情境；當發現所採取之一連串的行為無法呼應此長期的活動(Activity)時，則會返回進行調整。

3. Operations是人們反覆執行行為的操作(Operations)，是根據操作的技巧越來越熟練後才會變成的，且類似日常例行的工作，像是下意識就會做出來的動作，且也要根據在執行行為時的情境做調整，若情境改變，則操作會返回至行為的階段；因此操作是屬於適應及執行的階段。

活動階層性架構是由Leontiev (1981)學者所提出，Leontiev認為此架構最主要是強調外在的實際活動如何反應出內在的認知，因此，此架構是以外在實際活動為主，並進而呼應出內在認知為何，如圖2.2.1所示：

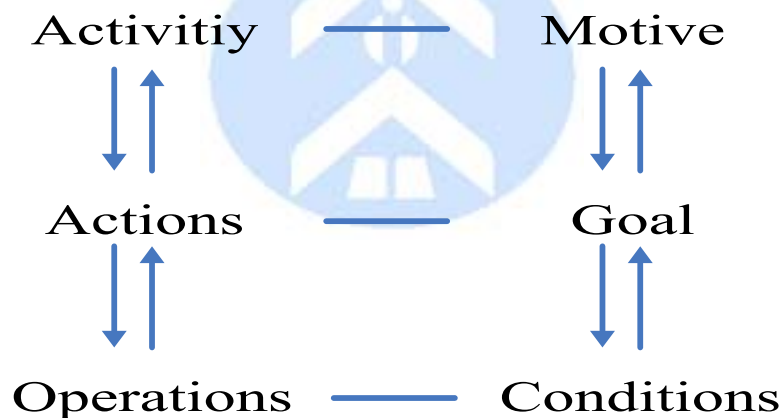


圖 2.2. 1 活動階層性架構 (Leontiev, 1981)

(二) 目標導向

所謂的目標導向為陳述活動理論裡3個物件，包括工具(Instruments/Tools)、對象(Subjects)、目標(Objects)，這三者須相互依賴、交流，意指這三個物件均為相互有關係，如圖2.2.2所示，對象必須透過工具來達到目標，最後轉換成預期的成效

(Outcome) (Engeström, 1987)。不過，Engeström認為只有三個物件(工具、對象、目標)的活動理論太過於簡單，沒有辦法符合人們在執行活動時的需求，因為人們的需求會一直變化，因此又發展出一套活動系統，從三個物件衍變為六個物件，這套活動系統強調人類的活動都是以對象與目標之間的關係為基礎的出發，這個關係是利用中介工具社群(Community)、角色(Division of Labour)，和規則(Rule)來協調，如圖2.2.3。

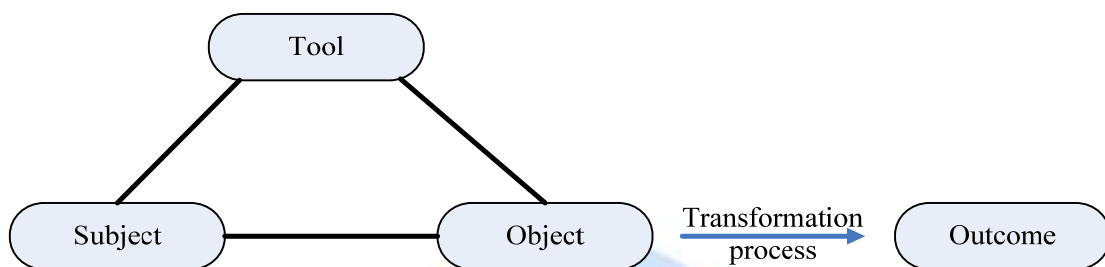


圖 2.2.2 個人層級的活動關係圖 (Engeström, 1987)

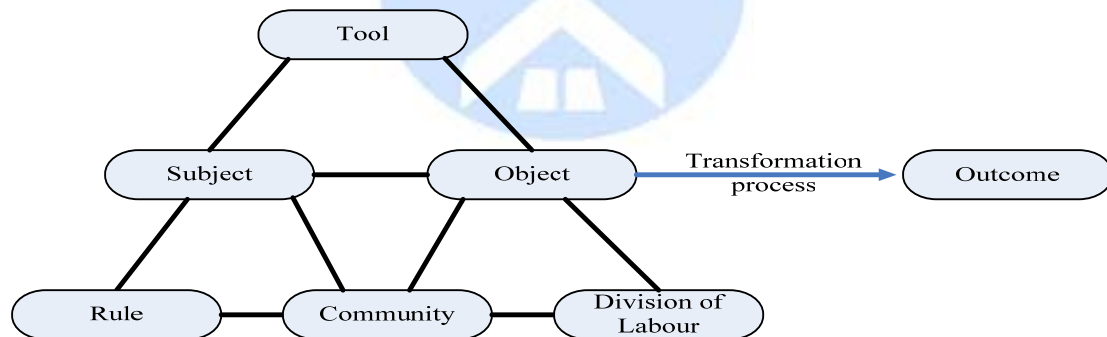


圖 2.2.3 活動系統 (Engeström, 1987)

如圖2.2.3所示，Engeström活動系統包括六個物件以及他們是如何互相影響進而達到預期的成效。六個物件分別為對象(Subject)，目標(Object)，規則(Rules)，工具(Instruments/Tools)，社群(Community)，角色(Division of Labor)，其中對象，目標及社群為直接的關係，要如何透過規則、工具、角色三者相互影響，因此規則、

工具、角色是屬於媒介(Mediation)，以下將介紹這六個物件及彼此是如何影響：

1. Subject：意指從事此活動的對象。
2. Object：意指所從事之活動的目標。
3. Instruments：意指對象從事活動時為了達到目標利用什麼工具去輔助並加以達成此活動的目的。
4. Community：意指從事活動的對象在哪個環境裡去執行的。
5. Rule：意指在社群環境裡，對象所執行的活動有什麼規則限制。
6. Division of Labor：意指在社群環境裡，對象在執行的活動裡扮演什麼角色。
7. Outcome：意指執行活動完的結果是否有達到預期的成效。

本研究探討Leontiev的活動階層性架構與Engeström的活動系統之間的相關性，運用學者Kuutti (1996)所提出的資訊科技(Information Technology, IT)觀點結合Leontiev的活動階層性架構與Engeström的活動系統，並對此兩者理論做一個相關性探討。如表2.2.1所示，左邊為Engeström的活動系統，包含六個物件：工具、目標、對象、規則、社群、角色；上方三個層級(操作、行為、活動)為Leontiev的活動階層性架構之外在執行的部份。此相關性探討的內容是以IT的例子表示，共有六個物件說明於三個層級中Leontiev的活動階層性架構與Engeström的活動系統之間的關聯。(一)工具—操作層級與工具的對照，所表現的是IT每天都會做的例程序，代表著操作的階段是屬於重覆的例程序；行為層級與工具的對照，所呈現的為利用工具來輔助所變化且執行的活動；活動層級與工具的對照，則是表達出想要使例程序自動化及建造一個新的工具；(二)目標—操作層級與目標的對照為提供跟目標相關的資料；行為層級與目標的對照為使目標能操作化；活動層級與目標的對照為讓某些事變成想要達到目標；(三)對象—操作層級與對象的對照為觸發已

做決定的反應；行為層級與對象的對照為讓對象在執行的活動裡所採取的行為有意義；活動層級與對象的對照為使對象對整個所執行的活動及目標能有效的學習及做出反應；(四)規則—操作層級與規則的對照為嵌入令人有深刻印象的規則；行為層級與規則的對照為使一連串的規則具有可見度及可理解度；活動層級與規則的對照為使規則之間能互相協議；(五)社群—操作層級與社群的對照為藉由連結很多人所一起執行的工作任務來建立一關係社群；行為層級與社群的對照為支援社群裡的所執行的活動及讓對象在社群裡的網絡能見度提升；活動層級與社群的對照為使社群能夠組成；(六)角色—操作層級與角色的對照為嵌入令人有深刻印象的角色；行為層級與角色的對照為讓工作的組織能具有可見度及可理解度；活動層級與角色的對照為使角色定位能重新整頓。從以上所有說明的例子可看出，活動層級到行為層級最後到操作層級皆表示著是從抽象的事物一直往下衍伸變越來越具體化、越細節。

表2.2.1 藉由資訊科技的觀點將外在執行層結合活動系統之分類表

	Operation-Level Support	Action-Level Support	Activity-Level Support
Tool/ Instrument	-automating routines	-supporting transformative and manipulative actions -making tools and procedures visible and comprehensible	-enabling the automation of a new routine or construction of a new tool
Object	-providing data about an object	-making an object manipulable	-enabling something to become a common object
Subject	-triggering pre-determined responses	-supporting sense-making actions within an activity	-supporting learning and reflection with respect to the whole object and activity
Rules	-embedding and imposing a certain set of rules	-making the set of rules visible and comprehensible	-enabling the negotiation of new rules
Community	-creating an implicit community by linking work tasks of several people together	-supporting communicative actions -making the network of subject visible	-enabling the formation of a new community
Division of Labour	-embedding and imposing a certain division of labour	-making the work organization visible and comprehensible	-enabling the reorganization of the division of labour

(資料來源：Kuutti, 1996)

綜合以上學者的觀點，本研究認為結合 Leontiev 的活動階層性架構與 Engeström 的活動系統，則可有效的分析出高階主管的認知層次架構與所需的知識內容。因此本研究運用活動理論將高階主管的心智流程明確的表達，接著透過本體論將高階主管所需的知識內容加以呈現。

(三) 內在與外在

活動理論有兩者不同的活動，分別為內在認知及外在執行，但是內在與外在通常不會分開來探討，因為內在認知是由外在執行所轉換的，外在執行也是由內在認知所轉換來的。內在認知說明人們心裡思考、思緒及規劃的事情，也就是所

謂的心智流程；外在執行則是將內在認知轉換成以實際的活動來執行。學者Kuutti (1996) 對於內在認知、外在執行分別舉三個簡單的例子來說明從外在執行如何呼應到內在認知，如表2.2.2所示，左方為活動層級、行為層級、操作層級。第一個例子為建立一個房子屬於活動層級，為了達到建立房子，行為層級所須做的事情為整理蓋房子用的屋頂材料及卡車運送磚塊，操作層級為了將這些屋頂材料及磚塊實踐於蓋房子，就必須開始敲打、設置這些材料以蓋好房子。第二個例子為完成專案屬於活動層級，為了完成專案，行為層級所採取的動作為編製程序模組，操作層級為了實踐模組的編製，必須真正去操作系統且寫出指令方法，並選擇適當的編碼指令來完成專案。最後一個例子為完成研究一個主題屬於活動層級，為了完成主題的研究探討，行為層級所採取的動作為搜集文獻、參加研討會及撰寫報告，操作層級為了順利的達成行為層級所做的事情，使用邏輯推斷及選擇適當的措辭用語以完成主題的研究。

表 2.2.2 活動、行為、操作三個層級的舉例說明

Activity Level	-Building a house	-Completing a software project	-Carrying out research into a topic
Actions Level	-Fixing the roofing -Transporting bricks by truck	-Programming a module -Arranging a meeting	-Searching for references -Participating in a conference -Writing a report
Operations Level	-Hammering -Changing gears when driving	-Using operating system commands -Selecting appropriate programming language constructs	-Using logical syllogisms -Selecting appropriate wording

(資料來源：Kuutti, 1996)

(四) 媒介

媒介，指的是Engeström活動系統包括的工具、規則、角色，因此媒介為強調人們活動的執行到完成必須透過媒介的影響才能達到，所以媒介在這裡扮演一個

非常重要的角色。舉例來說：對象須透過工具達成目標；在社群環境裡，對象所執行的活動要有什麼規則限制；在社群環境，對象在執行的活動裡扮演什麼角色，透過此三個媒介(工具、規則、角色)去相互影響對象、目標、社群及最後的成效。因此媒介在活動的發展或是執行期間會被創造或是被改變，因為人的需求會一直不斷變化，所以媒介的使用、設定、限制都是人們執行活動時一種知識的累積，而且媒介會影響人們外部行為及個人的心智流程。

活動理論目前被應用的領域包括HCI、系統設計流程(System Design Process)、資訊系統(Information System, IS)的開發等(Castro et al.,2005; Ashok and Beck, 2007; Freitas and Byrne, 2006)。HCI探討的活動理論主要是提供一個可以讓使用者與系統相互溝通的互動機制；活動理論應用於系統設計流程方面，主要為活動理論可視為一分析的工具，讓系統的設計流程可因應使用者的需求；資訊系統的開發所運用的活動理論，也跟HCI、系統設計流程大同小異，皆能針對系統的使用者需求而設計出不同的介面或是溝通機制。因此，本研究認為活動理論極其適合用來探討使用者需求，並根據其需求進而建立一溝通運作機制。

三、本體論

(一)本體論定義

本研究將運用本體論(Ontology)的概念分析並建置出具有一高階主管資訊知識內容的領域本體論。本體論一詞，是早先從哲學所分支下來的，但是近幾年已成為跟知識管理(Knowledge Management)、知識工程(Knowledge Engineering)相關的字詞。根據學者 Gruber (1993)所提出的定義，他認為本體論是一個共享概念化的形式說明。因此，本體論作為知識表達的基礎，可以避免重複的領域知識分析

且由於統一的術語和概念可達成知識共享的目的(阮明淑與溫達茂, 2002; Oscar et al., 2003)。本體論具有階層架構, 可以描述知識彼此之間的關聯, 能將較豐富的知識內容表達呈現。因此在某種程度上, 能夠讓領域本體論的知識內容具有一致性(Chandrasekaran et al., 1999; Noy and McGuinness; Vijayan and Storey, 2002)。

本體論被應用於許多領域, 主要由於可以明確詳述語意和關係的電腦標記語言(Extensible Markup Language, XML)表達能力。一般來說, 本體論的功用包括溝通(communication)、交互運作機制(inter-operability)、系統工程(system engineer)(Bunge, 1977; 阮明淑與溫達茂, 2002)。所謂的溝通是利用彼此之間共通的術語進行交流, 交互運作機制是無論在不同建構方法、模式、語言等等之間都有對照或翻譯的機制存在, 系統工程則是將本體論應用於系統工程, 將其具有的重複使用, 知識獲取, 可靠性及規範描述等優點達到知識再分享的用途。

因此本研究將會運用本體論分析高階主管所需的知識, 這些知識包括企業基本功能, 以呈現出知識相互之間的關係。

(二)本體論語言

本研究透過全球資訊網組織(World Wide Web Consortium, W3C)所制訂的本體論語言(Ontology Web Language, OWL)為基礎來建置成一知識本體論。OWL 是 W3C Web Ontology Working Group 於 2004 年所發表。OWL 最主要的目的是在於表示所有存在於網路中的文件與應用程式之間的相互關係, 因為文件與應用程式會因工具不同而使呈現出的方式不一樣, 因此 OWL 可以使網路上的內容轉換成不僅人們瞭解, 機器與應用程式也都明瞭, 進而達到語意網(Semantic Web)的一項新標準(張成儀, 2006; McGuinness and Harmelen, 2004)。

OWL 是從 XML、RDF (Resource Description Framework, RDF)及 RDF Schema

所延伸出來的，並參考 DAML (DARPA Agent Markup language)和 OIL (Ontology Interface Layer)所產生。圖 2.3.1 由 W3C 所提出的本體論語言架構。OWL 為敘述邏輯的概念演變而來的，在 OWL 中可分為三種等級的語言：OWL Full、OWL DL、OWL Lite (Matthew et al., 2004)。

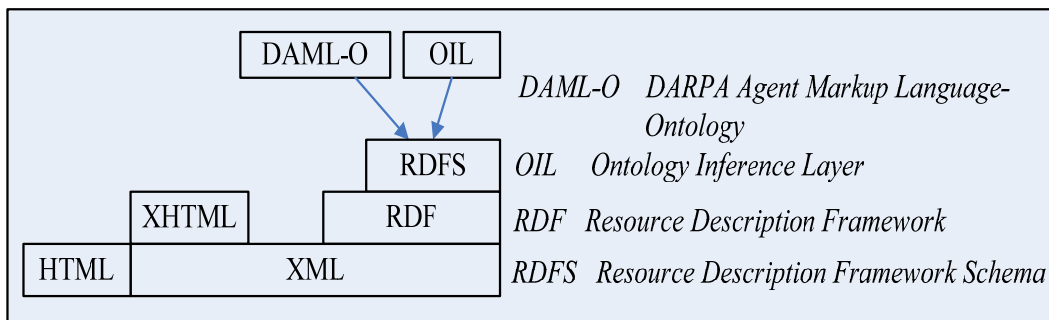


圖 2.3.1 本體論語言架構 (Fensel, 2000)

OWL 亦可描述專業或是一般性的領域知識，是目前最具表現力的本體論語言。OWL 相較其他本體論描述語言，所能支援的功能也較豐富以及表達能力較良好，因此，OWL 將成為未來本體論語言的主流。OWL 在知識的描述具有的優點包含，定義完善的語法 (a well-defined syntax)、正式的語意 (a formal semantics)、便利的表示 (convenience of expressive)、有效率的支援推論 (efficient reasoning support)、有足夠的表示能力 (sufficient expressive power)。

(三)本體論工程

本體論工程為建構本體論的方法，目前有諸多方法(Fernández et al.,1997; Gruninger and Fox, 1995; Uschold and King, 1995; Noy and McGuinness)皆是用來建構本體論，其中包括多倫多大學(Toronto University) Gruninger 等人所提出的 TOVE

本體論工程 (Toronto Virtual Enterprise, TOVE)，採用 TOVE 的原因為 TOVE 本體論工程是目前世界上極為重要的方法之一，由於它淺顯易懂，因此目前已廣泛被應用在許多建置領域本體論的過程之中 (Uschold and King, 1995; Kim, 2002; Fernández, 1999)。TOVE 本體論工程共分成六個階段，如圖 2.3.2 所示。

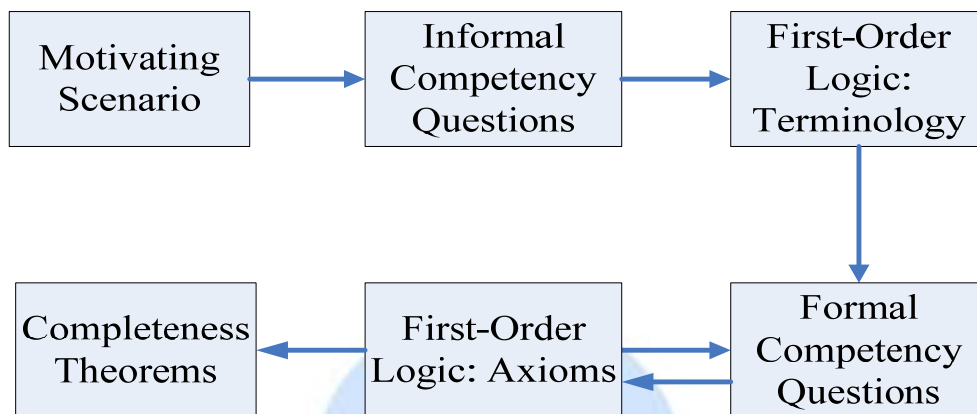


圖2.3. 2 TOVE 本體論工程六階段 (Gruninger, 1995)

1. Motivation Scenario

此階段的用意就是描述本體論所須解決的問題，及可能的解決方案。不論是重新定義本體論或是針對舊有的本體論加以延伸，motivation scenario 階段都扮演著重要的角色，靠著 motivation scenario 可使往後的維護人員瞭解設計本體論的用意，同時，它也將引導出評估整個本體論所須回答的問題—competency question.

2. Informal Competency Question

從 motivation scenario 可衍生出 competency question，它通常是一個員工在進行工作時會用到的需求。在本體論中，將 competency question 視為本體論的要求，

也是日後評估本體論的重要條件之一。此階段 competency question 是採自然語言的方式表示。一個理想的 competency question 應具備階層式的關聯，也就是越後面的 competency question 將會使用到先前 competency question 所回答出來的答案。同時，competency question 並不會直接產生本體論，它是用來評估本體論，因此，在設計本體論時應以能回答 competency question 為目標來進行設計。

3. First-Order Logic – Terminology

Terminology 階段須定義所有本體論會用到的詞組與關聯，以 TOVE 本體論工程來說，他們建議採 First-Order Logic 表示所有的詞組與關聯，同樣也可採用其相等的邏輯表示法，如知識交換格式 (Knowledge Interchange Format) 等。定義 Terminology 的第一步是先定義所有本體論會用到的詞組，同時這些詞組在邏輯表示法下是以常數或變數的方式來定義，接下來則是定義本體論的屬性及關，在邏輯表示法下則可用述語(Predicate)表示。

4. Formal Competency Question

在有了 Terminology 中的詞組後，這些詞組將 Informal Competency Question 轉成 Formal Competency Question。在這個階段中，Formal Competency Question 將成為最終評估本體論的準則，同時，靠著 Formal Competency Question 也可區分不同本體論的能力。

5. First-Order Logic – Axiom

Axiom 階段明確定義本體論的 Axiom，即限制或詞組的語意(包括邏輯關係)，TOVE 本體論工程於 Axiom 階段也是採用 First-Order Logic 進行描述。Axiom 階段是整個本體論建構過程中相當重要的階段，因為在定義本體論時，前幾個階段定義本體論的詞組 (包括述語)、階層關聯等，但這些定義並不能完全表達詞組中較

複雜的語意或限制。因此，在 Axiom 階段會將前幾個階段未完整定義的語意加以定義。

6. Completeness Theorem

Completeness Theorem 是利用 Competency Question 來評估所建立的 Ontology 是否符合第一階段的動機，也就是將真實的資料輸入進去，並利用推論機制來評估是否每一個 Competency Question 都能被正確地回答。

四、Protégé

Protégé 為一知識庫平台，由美國史丹佛大學 SMI (Stanford Medical Informatics) 中心以 Java 所研究開發的知識庫自由軟體，其知識庫最大的特色是應用本體論的觀念設計，提供了 Frame-based 的知識表達方式與一個開發推論引擎的環境，並納入 OKBC (Open Knowledge Base Connectivity) 模組，讓知識工作者可以建構一個以本體論為基礎的跨平台知識管理系統。Protégé 是目前世界上被使用來建置知識庫中最重要的平台之一，也是目前全球支援本體論的平台中最為廣泛使用的一個(葉文權, 2004; Noy and McGuinness; Noy et al., 2000; Grosso et al., 1999)。

(一) Protégé 的特色

Protégé 主要功能是輔助知識工作者建置知識，並將知識擷取的過程中所會遇到的障礙降為最低，進而建立一個以本體論為基礎的知識管理系統。Protégé 的架構分成 User Interface、Core Protégé 及 Persistent Storage 三個層級，如圖 2.4.1 所示：

1. User Interface

利用介面模組，包括 Classes、Slots、Forms、Instances 與 Queries，提供使用

者檢視及編輯知識庫，此外也可以利用 Plug-in 增加新的功能介面（Tabs 與 Widgets）。

2. Core Protégé

Core Protégé 為整個 Protégé 的核心，最主要是將使用者介面與知識庫做一個結合，並處理所有和知識庫維護有關的項目。

3. Persistent Storage

包括實體儲存媒體和 Knowledge-base Mapping 兩部份，利用 Knowledge-base Mapping 將本體論的實例(Instances)對映到實體儲存媒體上，以利未來系統的運作。

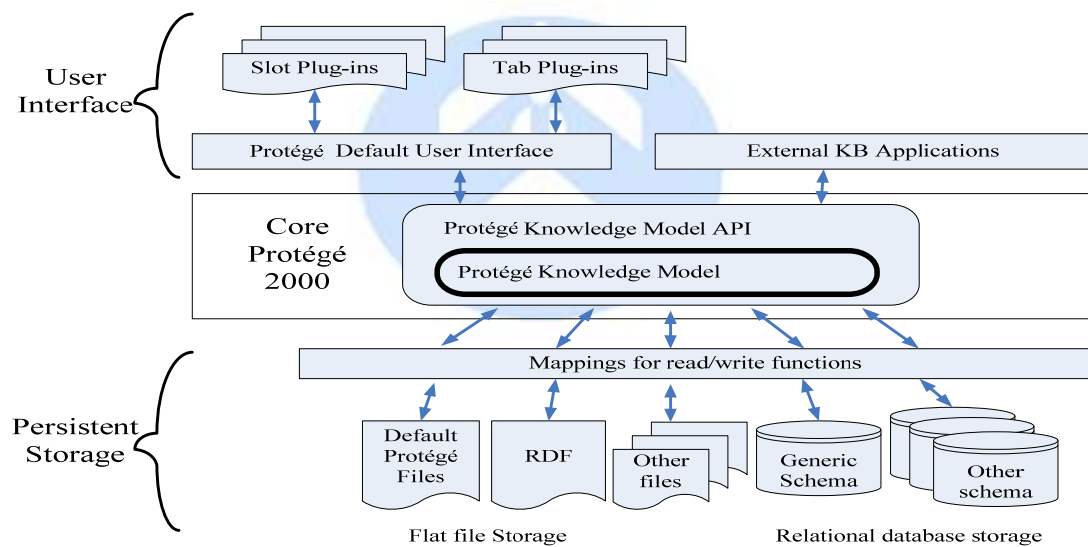


圖 2.4.1 Protégé 的系統架構 (Musen et al., 2003)

根據圖 2.4.1 可將 Protégé 的特點細分成以下幾點，包括：

- (1) 可擴充知識模組：根據自己的需求使用者可以自行定義與擴充領域知識。
- (2) 客製化使用者介面：可將介面進行客製化，能更有效的擷取知識。

- (3) 與其他應用系統的整合：運用 Protégé 與其他應用系統加以整合，使系統之間都能相互溝通，進而達到知識分享的目的。
- (4) Protégé 提供 First Order Logic(FOL)為基礎的編輯程式語言 Protégé Axiom Language(PAL)，方便使用者實作本體論工程(TOVE)之中的 Axiom。
- (5) 圖形化介面：輔助知識工作者使用其知識模組來建構領域知識本體論，進而降低知識擷取的困難。

(二) Protégé 的知識模組

Protégé 的知識模組，以 Frame-based 為基礎去呈現，是在做知識表達時常用的一個選擇方法(Karp, 1993)，其知識概念呈現與分類的型式被認為是與人類最相似的。在 Protégé 中的 Frames-based 包含了有 Class、Instance 與 Slot，其中 Class 就是所謂的類別，表示某領域的知識概念。Instance 為實例，亦即 Class 的實作。Slot（或稱為 Property）則用來描述概念的屬性或關聯，Slot 被定義出來之後，就可以被 Class 或者 Instance 引用。Facet 是 Slots 的限制條件，可說是 Slots 的屬性，用來定義 Slots 的功能，讓 Slots 有所限制。本體論的最重要的地方即在於各個知識概念類別如何的呈現(Noy and McGuinness)。

1. 類別(Class) 與實例(Instance)

類別是一個階層架構所組成，上、下階層的關係為本體論中的「is-a 或 a-kind-of」關係，亦即父類別與子類別。當子類別集合在一起，共同組成一個父類別，這就呈現了「part-whole 或 a-part-of」關係。實例是類別的概念實作，屬於類別，也繼承其所屬類別的所有屬性。

2. 屬性(Slot)

屬性用以描述類別或實例的屬性，在 Protégé 中建立的屬性，可重覆使用於具

有相同屬性的概念類別中，經由 Slot template 裡的 Domain 欄位即可檢視該屬性被應用於哪些類別之中，此舉可大幅提昇屬性的可重用性與可擴充性。由於 Protégé 相當具有彈性，可以將某個特定知識概念相關的知識屬性都鏈結在一起，維持其完整性與可瞭解性。

3. 規則限制(Facet)

Facet 為定義屬性的限制，如可以指定屬性的範圍大小、資料型態為何 (Boolean、Class、Float、Instance、Integer、String、Symbol)，若資料型態為數值，則可定義屬性的最大值及最小值，。



參、高階主管知識分析

高階主管工作環境為高複雜性，所要達到的工作責任不只一個。高階主管所需揹負的工作責任越多，所需提供的知識亦也會跟著增加，然而高階主管為了達到工作責任，必須將所有的工作責任之間的關係環環相扣，也因如此，導致背後所需的知識層面亦會跟著變動，在如此循環的過程之下，高階主管的知識需求乃成為變動的性質。本研究將高階主管工作責任分為追求利潤、提升競爭能力及永續經營。為了達到工作責任，必須有知識層面予以輔助，這些知識是為了提供高階主管適時且重要的資訊。因此，本章節探討如何結合本體論與活動理論，並進而建構一高階主管知識系統。本研究在此部份將藉由一家公司之高階主管實際所需的知識需求，建立其一般高階主管的知識分析與心智流程。本章節詳細的步驟包括：

- 一、透過對公司各部門負責人員的訪談來瞭解高階主管追求的知識為何，並分析出高階主管決策所需知識層面；
- 二、運用活動理論的活動階層性架構分析高階主管心智流程及認知層次架構；
- 三、運用本體論建置一高階主管經營公司所需的知識領域本體論。

以下將逐一詳細各步驟：

一、公司知識內容分析

本研究將探討一家公司的高階主管經營企業的情況來做為高階主管的需求分析。此公司位於南部，屬於製造業，產品大類為水泥、混凝土等，公司實際情況為各部門於每週會議要向總經理彙報當週的營運狀況，各部門分別為「行銷部門」、「資訊部門」、「生產部門」、「財務部門」。「行銷部門」負責每週營業處之北區、高屏區、花東區、雲嘉南區的銷售情況、同業資訊及市場狀況；「資訊部門」負責每週營業處之機器設備及維護的狀況；「生產部門」負責各營業處之生產線的生產狀況、原物料供貨來源與同業資訊；「財務部門」負責各營業處之營收總額、營收成長及獲利率。以下將對各部門於每週向總經理報告的實際狀況做一說明：

(一)「行銷部門」：行銷部門最主要是彙報各區域的銷售情形及產品是否有達到預計的出貨量以及根據銷售情形來決定生產量的多寡，還有各同業之間的資訊與行銷市場之狀況說明。當總經理聽完行銷部門此週的狀況時，發現銷售量為上升的情況，立即聯想是與成本管控、生產量的多寡及客戶的購買量有關。可是行銷部門的人員只能提供銷售狀況及產品量變動的情形，並不能將總經理腦海裡所聯想到的知識呈現出來。所以，行銷部門總是無法即時的提供總經理相關的資訊，必須透過人員再花時間將總經理想知道的知識找出來，沒有辦法讓總經理順著他腦海裡的想法走，因為彙報人員只能提供行銷部門自己的資料，不能與其他部門做整合，所呈現的資訊也跟著有限制。有了以上的問題，總經理覺得企業營運裡應該要能跟其他部門的知識做個結合，才能適時的找出重要的資訊及做出適當的決策。因此企業營運的知識裡與行銷部門彙報的情形相關的知識為：銷售、成本、客戶、生產。

(二)「資訊部門」：資訊部門主要是將公司各營業處營運所需的資源，像是機器設

備及工廠維護等狀況。總經理聽完資訊部門人員的彙報，發現到機器設備運作情況比以往下降，因此聯想出機器設備運作下降可能與機器設備的維護狀況或是成本管控有關。然而，資訊部門的人員就必須找出與機器設備運作下降有關的知識，不過總是需要花更多時間將這些知識找出來，因為這些知識不是歸屬於資訊部門，並不能即時提供總經理這些相關知識。因此，與資訊部門相關的企業營運知識是包括：成本。

(三)「生產部門」：生產部門是負責彙報每週公司的生產狀況與同業資訊，是否有達到預計的生產量，及原物料每週的成本、價格及供貨來源是否有所變動。總經理聽完生產部門的彙報後，發現到生產量上升，聯想出是與成本管控，產品銷售量有關。不過，生產部門的人員必須花費時間再將這些成本、產品、銷售的知識找出，因為這是屬於其他部門的知識，並沒有與生產部門的知識做整合，無法即時提供知識給總經理讓他能進行比較。因此，與生產部門相關的企業營運知識包括：成本、產品、銷售。

(四)「財務部門」：財務部門為每週彙報各營業處之營收總額、營收成長及獲利率。當總經理聽完彙報後，發現營收總額比以往下降，聯想可能是與成本管控、銷售量的多寡、生產量下降有關。但是該部門人員卻無法即時提供成本、銷售、生產的知識讓總經理一起比較。因此，與財務部門相關的企業營運知識則包括：產品、銷售、生產。

根據以上的敘述，本研究將高階主管之工作責任及所需要的知識分析整理成表 3.1.1。表 3.1.1 著重於高階主管經營公司時所需的主要考量，其分別為高階主管所需的管理營運認知層次作探討，及建立高階主管心智流程與運作所需的相關的知識需求及內容。這些知識內容為各認知層次高階主管面臨各工作責任所需的知

識內容，分為主要知識與次要知識，主要知識為高階主管為了達到三項工作責任時，直接想到的知識層面，即為主要知識；次要知識為所直接想到的知識層面，之後聯想到的第二、第三甚至更多的知識層面皆稱為次要知識。表 3.1.1 所描述的工作責任分為三種，每一種工作責任各有其主要知識及次要知識；高階主管為了達到追求利潤，所直接想到的知識為銷售情況，當看完銷售情況，聯想到銷售情況是與成本的控管相關，成本的控管又取決於生產情況，生產的情況也必須依照客戶的需求來配合；提升競爭能力主要知識為成本，因為成本的高低對於提升競爭能力是一大助力，高階主管欲達到提升競爭能力所直接想到的知識為成本，瞭解完成本，聯想產品銷售的情況可能與成本相關，也須觀察競爭者的產品與公司的是否相似，產品的銷售更與生產的情況息息相關；永續經營主要知識為產品，因為公司必須一直開發出好的產品才能讓公司生存發展，有了好的產品就能吸引客戶來購買，產品的銷售情況與成本控管也有關係，要吸引客戶購買產品更要瞭解客戶所在的區域及產品是否能滿足客戶的需求。

表 3.1.1 管理營運的認知層次所需的知識內容

工作責任	主要知識	次要知識
追求利潤	銷售	成本、生產、客戶
提升競爭能力	成本	產品、競爭者、生產
永續經營	產品	客戶、成本、環境

高階主管知識需求分為主要知識及次要知識，但為了滿足其高階主管認知層次架構，本研究須將所有的知識內容予以彙整，讓高階主管在存取知識時，能快速的找出他所要的知識內容是屬於哪個知識面，也能根據所彙整的知識層面因應

高階主管的知識需求變動，其意指能透過直接的主要知識及聯想的次要知識給予高階主管最適當的知識，並能即時完全的搜尋出符合其知識需求。因此，本研究將主要知識及次要知識裡的知識內容，根據訪談完公司所需要的知識面予以歸納，即可將高階主管所需的知識需求分為六大知識面，分別為：

1. 競爭知識面。
2. 行銷知識面。
3. 客戶知識面。
4. 環境知識面。
5. 成本知識面。
6. 生產知識面。

六大知識面與高階主管管理營運的認知層次及所需知識內容可結合為表 3.1.2。表 3.1.2 所敘述的六大知識面分別滿足管理營運的認知層次裡的追求利潤、提升競爭能力、永續經營，這六大知識面裡也包括了高階主管企業營運時所需的相關知識內容。

表 3.1.2 知識面與認知層次及知識內容相關表

六大知識面	管理營運的認知層次	所需知識內容
競爭面	提升競爭能力	競爭者本身的產品價格、銷售數量、生產成本、原物料成本、機器設備成本、銷售額、比較市場狀況
行銷面	追求利潤	公司在銷售方面的知識，具備市場狀況(產品價格、銷售數量)、產品、利潤、生產成本、原物料成本、庫存盤點、配送
客戶面	永續經營	公司所擁有的客戶及銷售狀況的知識，應當具備了客戶資料及客戶相關的信用評等、銷售額、銷售數量、產品、配送
環境面	永續經營	公司營運的據點區域的知識，具備了季節、氣候、銷售據點、製造廠房、市場狀況、未來市場需求預測
成本面	提升競爭能力	整體的銷售狀況的知識，具備了人事成本、原物料成本、機器設備成本、生產成本、儲運成本、銷售成本、產品價格、利潤、稅率、營收成長、獲利率、匯率
生產面	提升競爭能力	公司的生產狀況，是否有達到預計的生產量，及原物料每週的成本、價格及供貨來源是否有所變動、產品銷售量、生產成本

二、高階主管認知層次分析

根據高階主管在經營企業時所遇到的實際狀況與瞭解完高階主管所需要的知識內容，本研究將運用活動理論，學者 Leontiev(1981)所提出的活動階層性架構(如圖 2.2.1)的概念，將高階主管的管理營運的認知層次加以分析，進而建立其高階主管的心智流程。從高階主管的心智流程，如圖 3.2.1 所示，可以分為四個階層，第一階層為高階主管的工作責任，第二階層為高階主管經營企業所需的六大知識面，第三層為六大知識面的知識來源，第四階層為高階主管經營企業所面臨的工作環境，這四個階層是由內在認知所形成的過程來顯現外在執行所呈現的形勢。

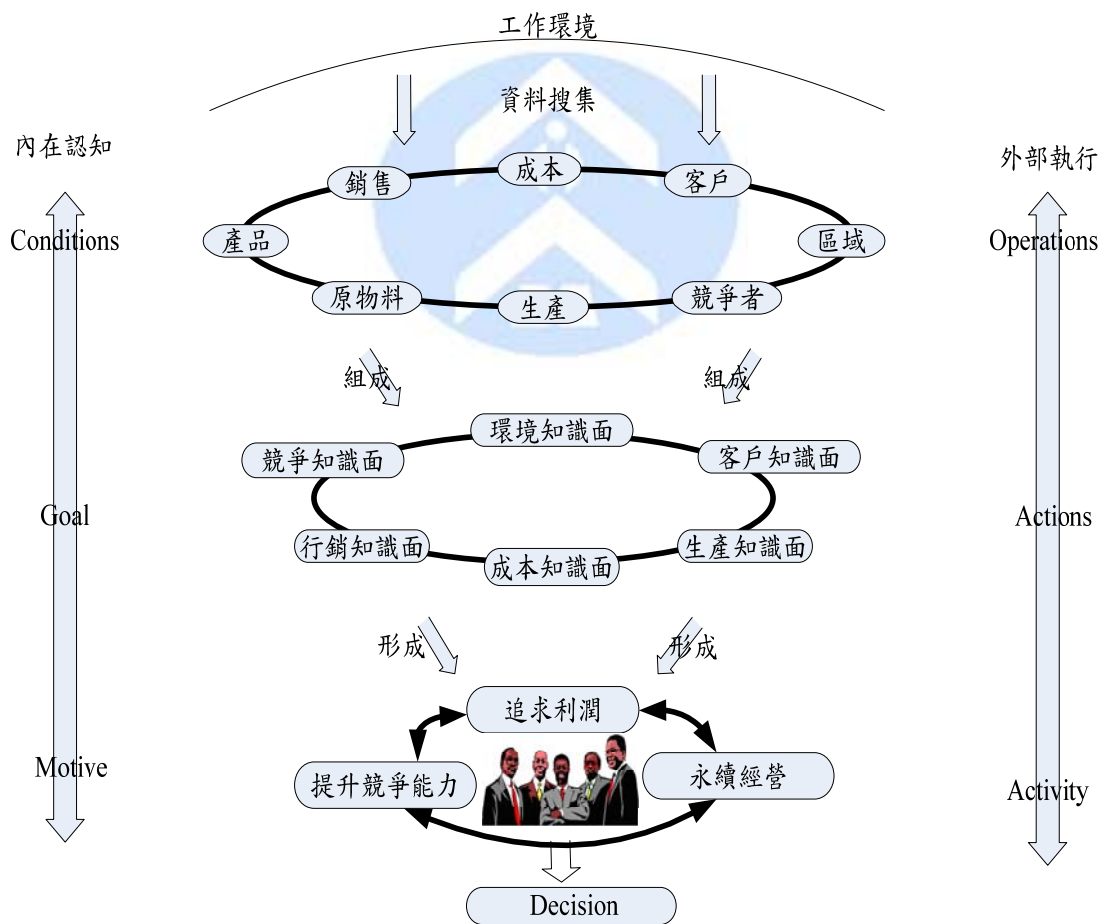


圖 3.2.1 活動理論與高階主管決策知識心智流程圖

(一)高階主管的工作責任：

高階主管所揹負的工作責任為追求利潤、提升競爭能力、永續經營，這三項工作責任是企業為了能生存發展所必須具備的，因此必須是環環相扣。當高階主管想滿足追求利潤的責任，會先思考什麼知識能滿足追求利潤的責任，當此知識可以滿足此責任時，可能又要思考此知識是否可以滿足其他兩個工作責任，在這樣子的循環聯想之下，高階主管的三大工作責任必須緊密的連結著，才能為公司做出良好的決策。三項工作責任分別代表的意義如下：追求利潤最重要的當然是幫企業做到利潤最大化的目標，因為經營企業想當然爾是要為公司謀利潤，創造出最大盈利；提升競爭能力也是經營企業重要的一環，因為企業除了追求利潤之外，也必須想出能為公司提升競爭能力的策略，這樣才能繼續生存發展下去，因此，這些策略不外乎是整合資源、良好品質的產品等等；最後，最重要的是，讓企業能永續經營，但是該如何讓公司永續經營才能讓公司持續生存，在此，企業必須要能為客戶創造價值、允諾企業社會責任，所謂的企業社會責任主要是在探討環境保護這方面的議題，當公司能做到企業所應揹負的社會責任時，也代表著對顧客的承諾，這樣就能一直創造新的顧客，為公司達到永續經營的目的。但是，為了達到此三項工作責任，則必須要有公司營運所應有的知識層面，這些知識層面是為了要輔助並形成高階主管能滿足他的工作責任。當此三項工作責任均能達成時，高階主管對於企業決策也能更快速的做決定，讓公司能於最快速且最恰當的時機下解決所面臨的問題。

(二)高階主管經營企業所需的知識面：

為了讓高階主管滿足三大工作責任，必須要提供高階主管經營企業所需的知識功能，本研究將企業所需的知識功能分為六大知識面，此六大知識面包括了成本

知識面、生產知識面、行銷知識面、客戶知識面、競爭知識面及環境知識面。這六大知識面分別為三大工作責任的主要知識及次要知識，這些主要知識及次要知識都是用來輔助高階主管，進而做出決策。但是，經營企業，高階主管必須要擁有該具備的知識，這些知識就是本研究所劃分的六大知識面。主要知識所指的是，舉例來說，當高階主管想達到追求利潤這個工作責任時，第一個所想到的知識面為行銷知識面，此行銷知識面就稱為主要知識；但是，高階主管看完行銷知識面之後，可能會因為這個行銷知識面而聯想到下一個知識面，如成本知識面、生產知識面及客戶知識面，因此，成本知識面與生產知識面就是次要知識。依照三大工作責任劃分，對於追求利潤而言，它的主要知識為行銷；次要知識為生產、成本、客戶。至於提升競爭能力主要知識為成本；次要知識為產品、競爭、生產。最後永續經營主要知識為產品；次要知識為客戶、成本、環境。

(三) 六大知識面的知識內容：

上述第二階層所提到的六大知識面，是用來提供高階主管達到工作責任，以利做出決策。但是，六大知識面裡又是以哪些知識內容所組成，因此，第三階層所要探討的是六大知識面各分別有哪些知識內容將知識面所組成。

1. 成本知識面：成本知識面包括的知識為成本、生產、產品、銷售。
2. 行銷知識面：行銷知識面包括的知識為銷售、生產、區域、產品、客戶、競爭者。
3. 生產知識面：生產知識面包括的知識為生產、成本、產品、銷售。
4. 客戶知識面：客戶知識面包括的知識為客戶、產品、區域。
5. 競爭知識面：競爭知識面包括的知識為競爭者、產品、成本。
6. 環境知識面：環境知識面包括的知識為區域、客戶、銷售。

以上六大知識面所包含的知識內容，都是第三階層提供至第二階層以組成的六大知識面。此知識內容為提供高階主管經營企業若面臨到問題時，能快速且有效的知道問題所在，並進而將對的知識給予高階主管，讓高階主管及時的做出決策。

(四)高階主管經營企業所面臨的工作環境：

第四階層為經營企業時，高階主管所面臨的工作環境，透過公司員工從事工作任務時，必須與其他部門或是外部資訊交流、相互溝通，也因為透過這樣的溝通，才能將知識萃取及分類，圖 3.2.1 可看到工作環境透過資料搜集，才可以將公司人員所溝通搜集的資料予以分類，並一一歸屬於該專屬的知識面裡。雖然知識的萃取可以從這樣的大環境裡取得，但是工作環境是屬於非常複雜的一環，因為企業每天所面臨的問題是如此的多，若不能透過資料搜集，高階主管該如何從中獲得重要的資訊並加以運用，因此，若能在大環境中，透過良好的工具及技術取得有用的資訊，那麼高階主管則能順利的解決經營企業所面臨的問題。

(五)內在認知的過程及外在執行的顯現：

圖 3.2.1 之內在認知及外在執行為運用學者 Leontiev (1981)所提出的活動階層性架構裡的概念，將內在認知的過程及外在執行的顯現應用於高階主管的心智流程，內在認知為高階主管為了達到三項工作責任，心裡所思考的過程。因此，本研究透過內在認知反應呈現之後，高階主管必須將此內在認知轉變為外在執行顯現的部份；在外在執行顯現的部份，高階主管必須將內在思考的過程與外在執行相互配合予以呈現。舉例來說：當公司在供貨需求皆滿足的狀況下，可以為公司追求利潤，並達到利潤最大化目標，但是必須透過實際的活動來達成此目標，因此，必須要採取市場定位的活動，但是市場定位又需要一連串的活動來達成，這

裡就包括了分析客戶需求等等的活動，最後就如第三階層所提到的在工作環境裡透過資料搜集的動作，將資訊搜集、彙整後，找出符合追求利潤的資料。因此，從內在認知及外在執行能瞭解高階主管揹負的工作責任到最外層的工作大環境裡，是從抽象的概念變成實體化，到了最外層的大環境，將是公司人員去搜集資料、彙整，因此一直延伸，將可看出資料一直被往上傳遞，並經過彙整、歸納、組成，最後滿足高階主管的工作責任，進而做出適當的決策。本研究運用此階層性架構最主要的目的是為了能解決以往高階主管在做決策時，無法有效的集中知識及部門人員亦無法將高階主管所聯想到的知識予以即時提供，讓高階主管在面臨問題時，常常因為沒有足夠的知識層面來輔助他做出適當的決策，因此，本研究所建立出的心智流程決策圖是運用學者 Leontiev (1981)所提出的活動階層性架構，亦即內在認知與外在執行的概念，有了此一概念將可反應出高階主管心智的需求及外在所採取的活動是互相配合及呼應，最後才能把高階主管最需要的知識需求予以提供，進而做出決策及持續的生存發展。

根據本研究所分析的高階主管心智流程，若欲將高階主管所揹負的工作責任之認知層次架構與以分析，須運用學者 Leontiev(1981)所提出的活動階層性架構，此架構能將高階主管經營企業的實際情形，與高階主管的心智流程加以結合，才能將高階主管所應揹負工作責任的認知層次架構予以分析。

(一)追求利潤

追求利潤對高階主管而言所需要的主要知識為行銷；次要知識為成本、生產、客戶，所謂的主要知識是總經理在追求利潤的同時，第一個直接想到的即為主要知識，其次聯想到的則為次要知識。公司在面臨競爭市場的同時，為了達到追求利潤，將著重在企業內部的需求，也根據所生產的產品及其銷售情況及產品的成

本，與客戶資料做個詳細的瞭解，才能將內、外部需求顧慮周全，在這樣的條件之下，才能將所要達到的利潤，在內、外部詳細的佈局後，順利的完成追求利潤此項重要的目標。如圖 3.2.2 所示，內在認知為追求利潤此責任，為了實行利潤最大化，必須要先滿足供貨需求，才能反應出追求利潤此責任是否有達到；外在執行所表現的是高階主管為了要達到追求利潤，所採取的活動為市場定位，有了市場定位之後，則必須先分析客戶需求及瞭解各區域的文化為何才能進而達到市場區隔，最後從所分析出的客戶需求及運用成本、行銷、生產、客戶知識來提供高階主管，進而達到追求利此目標。

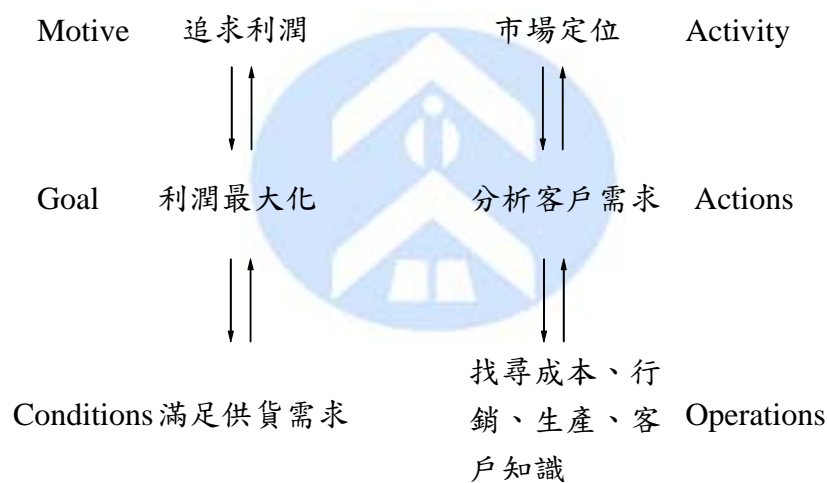


圖 3.2.2 追求利潤階層性架構

(二)提升競爭能力

此項提升競爭能力之工作責任，對高階主管而言所需要的主要知識為成本；次要知識為產品、競爭者、生產。企業提升競爭能力是為了加強自身的競爭優勢，因此，此公司將他們內部的生產力與所能利用的資源，無論是內部還是外部的資

源，都要強化及有效的運用，並積極的開發自身的核心能力，才能與競爭者抗衡，進而維持企業的競爭能力。如圖 3.2.3，透過此內在認知與外在執行之活動階層性架構，將高階主管的心智流程到實際的採取行動加以描述，內在認知為提升競爭能力，為了實踐提升競爭能力，必須要先強化企業自身的能力與生存發展，才能瞭解是否有達到提升競爭能力；外在執行為高階主管為了要提升競爭能力，所採取的活動為整合資源也就是將企業所運用的資源有效利用，然而整合資源需要降低企業的生產成本，才能將資源有效運用，因此為了降低生產成本，企業可從競爭者著手，觀察對方的生產數量、價格等等，最後從成本、產品、競爭者、生產等知識輔助高階主管，進而完成目標。

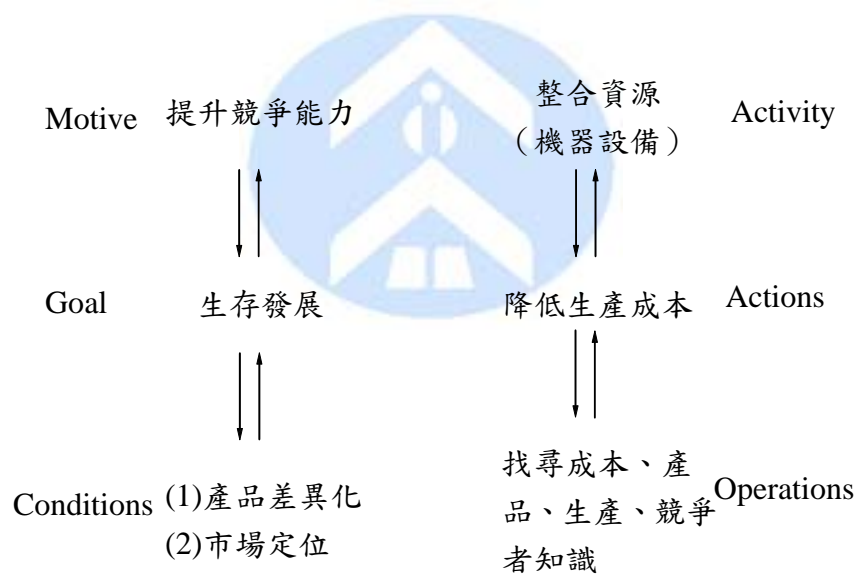


圖 3.2.3 提升競爭能力階層性架構

(三)永續經營

在永續經營方面，對高階主管而言所需要的主要知識為產品；次要知識為客戶、成本、環境。因為此公司在現今市場的強烈競爭下，必須要能夠維持生存，

經營者更要好好的思維出什麼樣的模式才是適合企業生存發展，因此實施企業社會責任，企業社會責任則是包括近幾年相當熱門的議題，就是必須要讓企業著重在環境保護這個課題及創造顧客，並促進附加價值。如圖 3.2.4 所示，內在認知為永續經營此責任，為了讓企業能永續經營，首先要極力的為客戶創造價值，所謂的創造價值就是要能瞭解客戶需求為何，進而為客戶創造價值，因此才能反應永續經營此責任是否有達到；外在執行可以看到的是高階主管為了要讓企業營運能永續經營，所採取的活動為實施企業社會責任，此活動已是各企業必須採取的策略，然而為了有效的達到企業社會責任，必須要讓企業著重在環境保護及創造顧客，也才能促進附加價值，為了要達到上述之要求，高階主管還需瞭解成本、客戶、產品、環境知識，因為有了這樣的知識提供，最後才能輔助並實現所謂的永續經營。



圖 3.2.4 永續經營階層性架構

三、高階主管知識內容分析

(一)語意網(Semantic Web)

運用活動理論可得知企業經營人在制定決策時所需思考的心智模式及說明高階主管應擔當的企業責任有三，追求利潤、競爭能力，和永續經營。除了瞭解高階主管所揹負的工作責任之外，高階主管還需瞭解企業基本的功能知識為何，才能滿足達到此三項責任之認知層次架構，最後幫助企業解決問題，進而做出決策。因此運用本體論分析以建立出具有高階主管知識內容的領域本體論，更運用語意圖 (Leung and Wong, 1990; Yang, 1990; Peters and Shrobe, 2003) 的方式來描述高階主管知識內容之間的關係，將原本具有複雜、龐大知識內容之間的關係所聯繫起來。有了此一領域本體論及運用語意圖，便可完整的將企業所應具備的功能知識內容相互之間的關係加以描繪，以提供一整體性的概觀，讓高階主管清楚明瞭企業各功能知識之間彼此所隱含的關係。如圖 3.3.1 Level 1 的企業功能知識語意圖所示，所連結的關係為第一層大類彼此之間的關係；圖 3.3.2 為本研究將領域知識本體論分為六個知識面：成本、生產、行銷、客戶、環境、競爭，這六大知識面，底下所包含的皆為各自的子類別；圖 3.3.3 完整功能知識語意圖是透過圖 3.3.1 與圖 3.3.2 衍伸而來的完整關係語意圖，此完整語意圖是將初級語意圖各個知識層面之間所擁有的關係予以聯繫，進而表現出其知識內容。

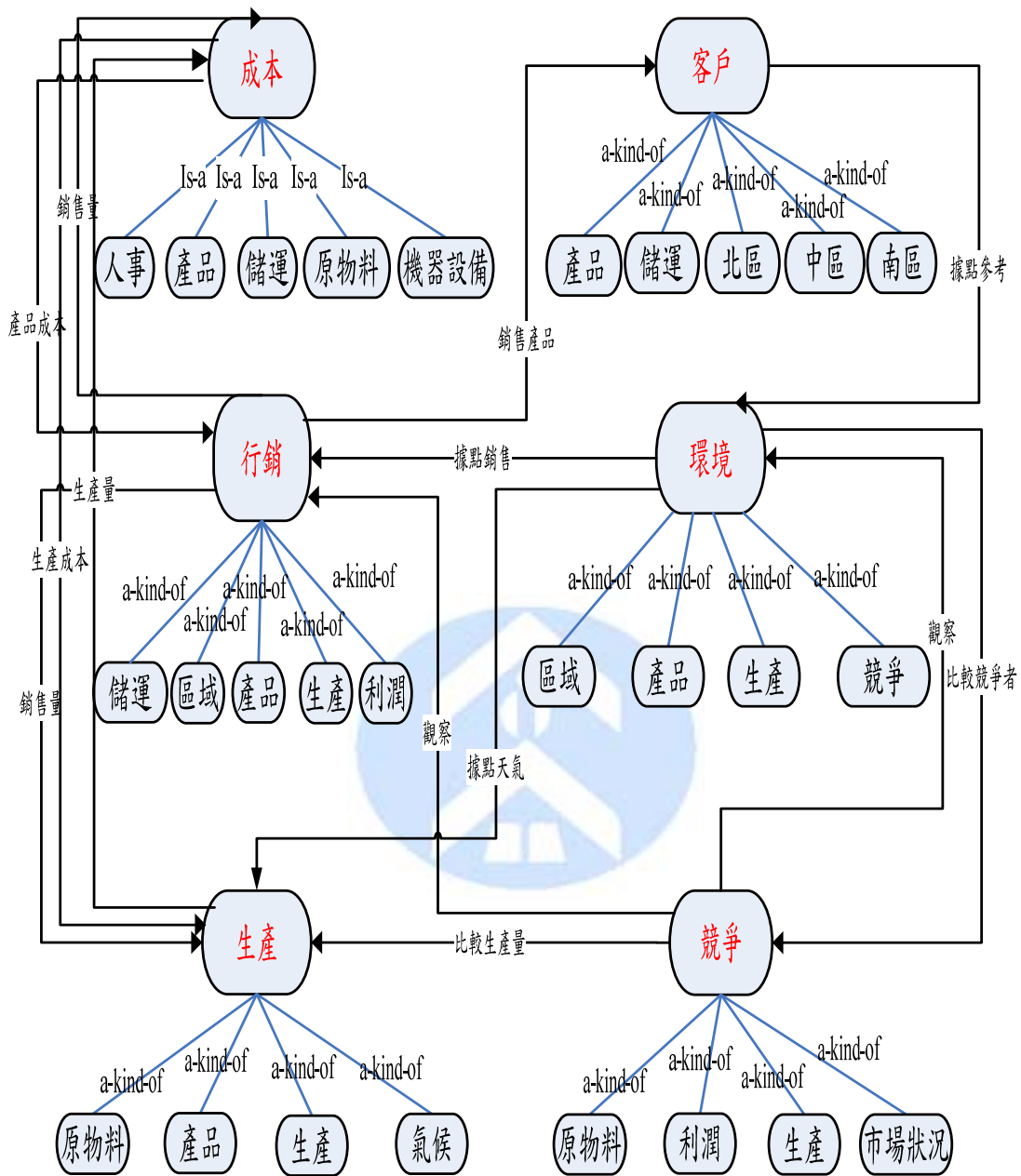


圖 3.3. 1 Level 1 企業功能知識語意圖

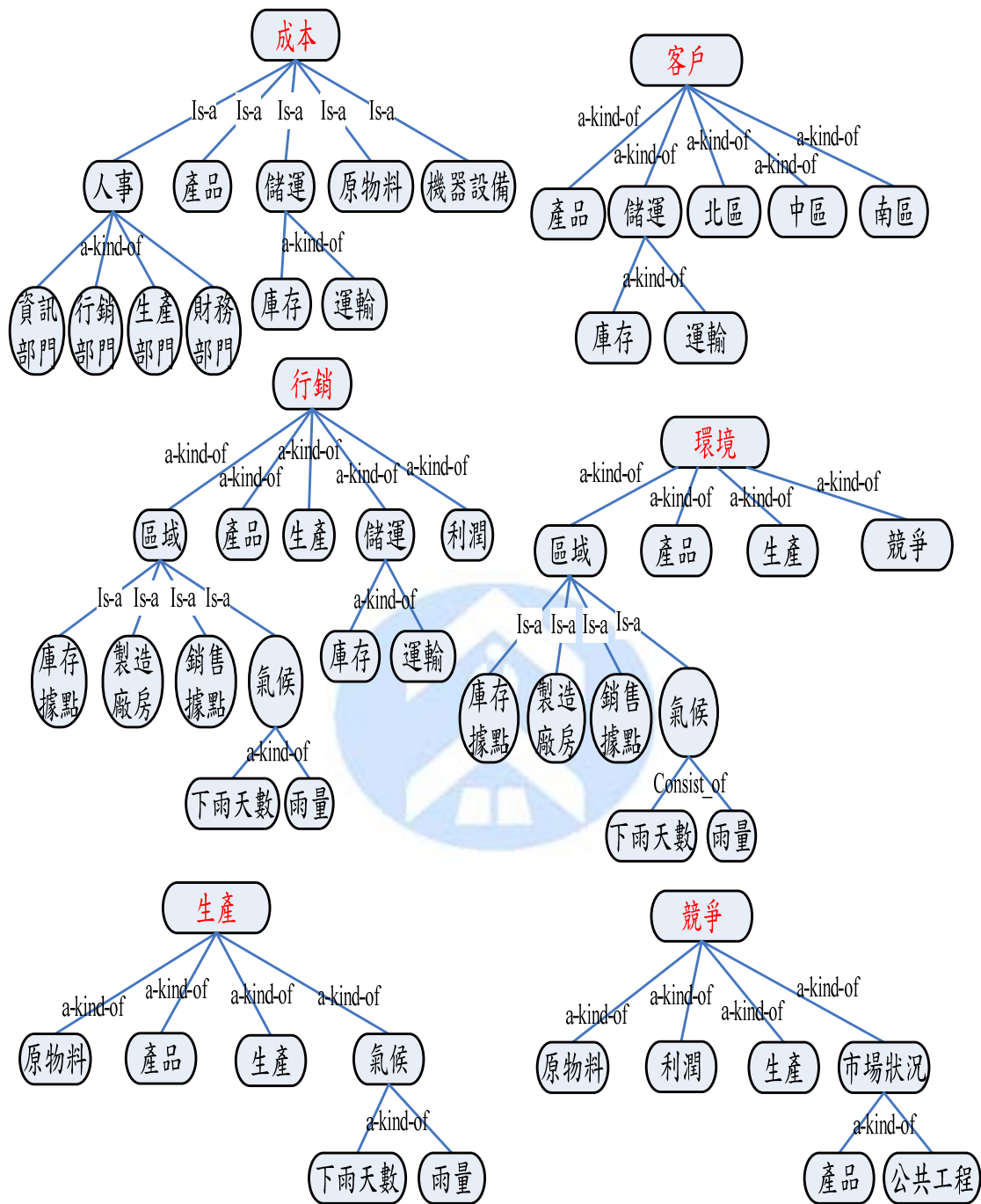


圖 3.3.2 企業各功能知識層次語意圖

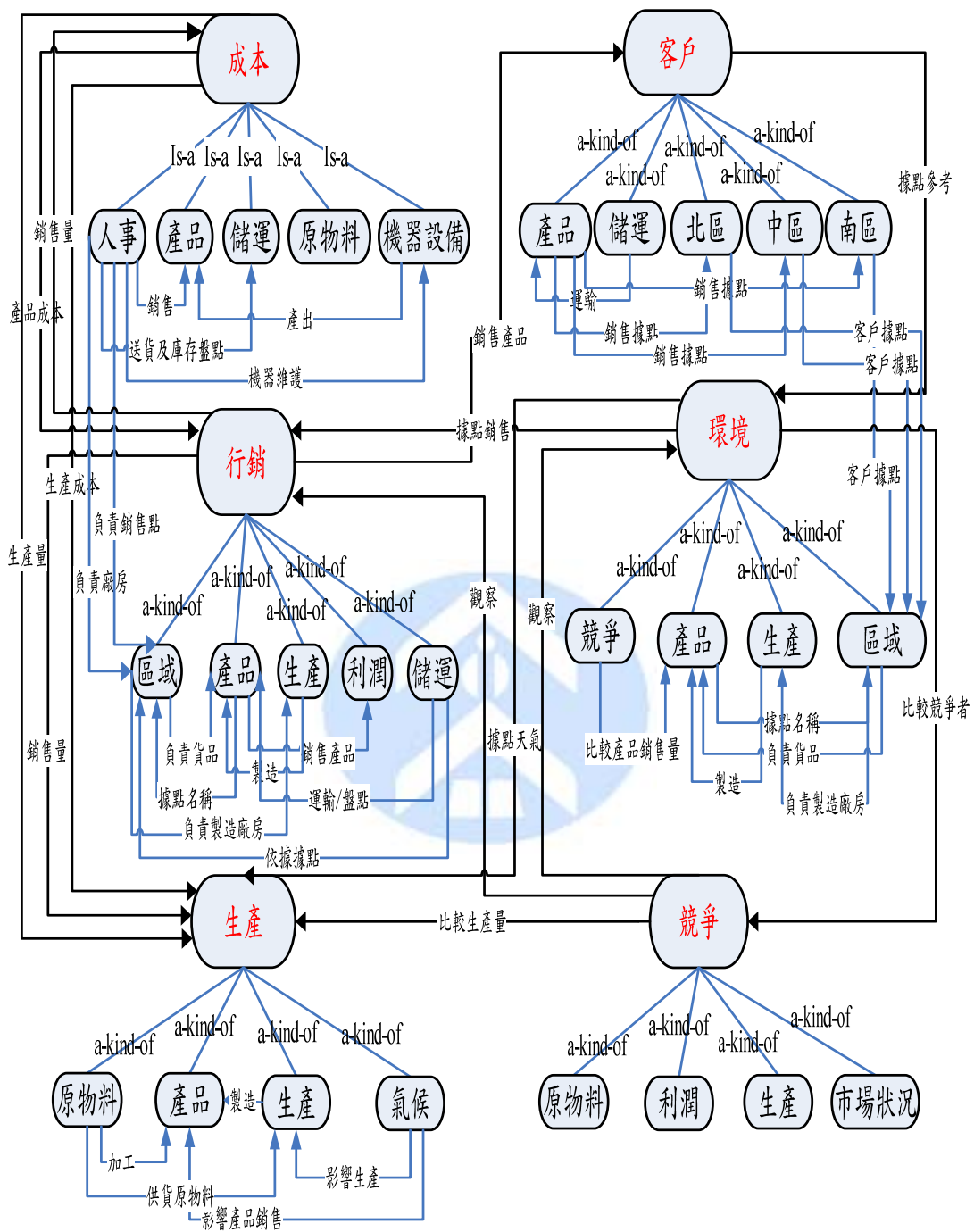


圖 3.3.3 企業功能知識內容完整語意圖

(二)類別的制定

高階主管的認知層次架構有三種，分為追求利潤、競爭能力，和永續經營，但這是屬於高階主管的心智流程，本研究在此方面已透過運用活動理論分析出高階主管的決策知識，在建立出領域知識本體論之前，必須先瞭解高階主管實際需求為何，知道了實際需求之後，再根據訪談公司的結果，將可得知此公司分為四個部門：行銷、資訊、生產、財務，再依此四個部門的劃分及歸納，本研究將高階主管的知識架構分為六大類，行銷部門包括產品的銷售、購買產品的客戶、產品銷售的區域及季節氣候的影響，以上皆可以歸納成三個知識面：行銷知識面、環境知識面及客戶知識面；資訊部門歸屬於成本知識面，因為資訊部門主要是機器設備及工廠維護情況，公司為了提升競爭能力必須有一知識面可幫助公司瞭解競爭者的情況，此知識面即為競爭知識面，例如競爭者之產品的銷售、成本的管控等；生產部門為生產知識面，生產知識面包括公司的生產狀況、原物料供貨來源等；財務部門則為成本知識面，將包括一切人事成本、產品成本、機器設備成本、儲運成本、銷售成本、生產成本、原物料成本等。以 OWL 建置此領域知識本體論將分為成本、行銷、客戶、競爭、環境、生產六個知識面為主，每一個知識皆分為主要知識及次要知識，讓高階主管可以更順利的瞭解面臨知識面時，哪個知識才是此知識面最主要的知識或是次要的知識，因此這六大知識面將可細分成如下，如圖 3.3.4 所示：



圖 3.3.4 OWL 建置之領域知識本體論

如圖 3.3.5 所示，關係屬性為本研究建置本體論類別與類別之間所須的屬性。關係屬性代表著知識概念彼此之間的關係，能連結知識物件。本研究建置的本體論之關係屬性，共包括了 20 個關係屬性(Object)，資料屬性(Datatype)有 48 個，此二個屬性將類別之間的關係與類別的描述加以說明。圖 3.3.5 原料加工物品、所需機器設備、維修人員，三個關係屬性透過彼此的連結，能夠快速的找出其相關知識，如圖 3.3.6，本研究將高階主管知識本體論以成本知識面底下的類別原物料為例，原物料類別，包括了 4 個實例，分別為土、水、火、鐵。透過土的知識內容裡可瞭解到土的原料加工成品為水泥，再透過水泥的知識內容裡可再深一層看到水泥的所需機器設備為挖土機，從挖土機又可更深入看到，挖土機的維修人員為紫瑜。從關係屬性的連結可以看到知識與知識之間的關係一直在衍伸，由此可知，本研究建置的本體論能透過此關係屬性將所有知識內容予以聯繫，呈現出一具有複雜關係的本體論。

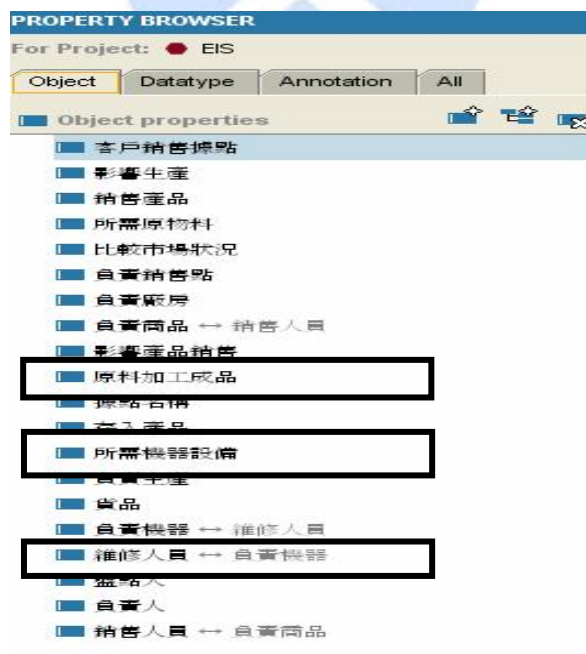


圖 3.3.5 關係屬性

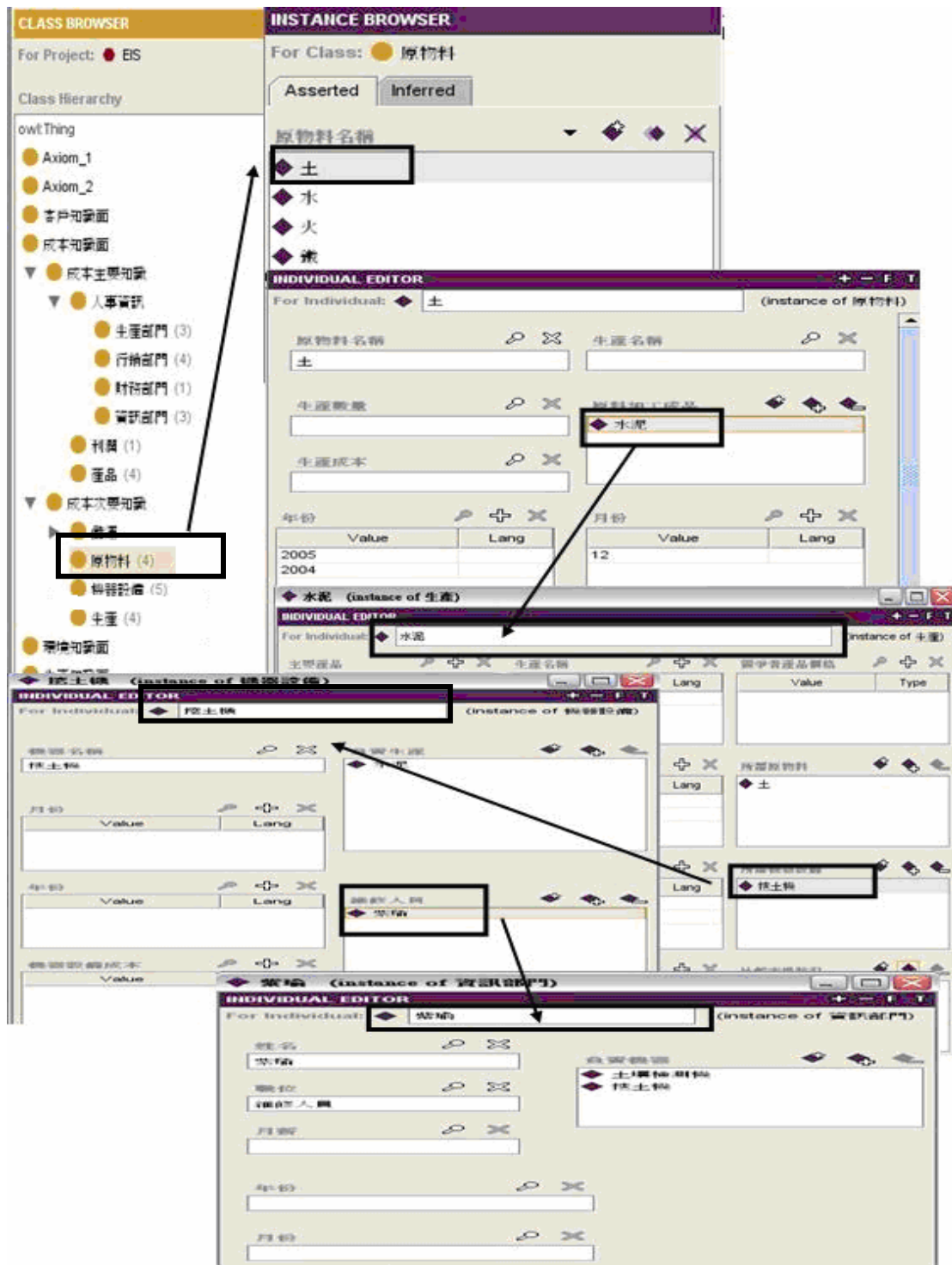


圖 3.3.6 知識連結關係本體論

肆、系統分析與設計

此章節將介紹本研究系統建置前分析的系統概念及根據系統概念進而設計的系統架構，及系統如何的建置及所有功能的運作。本研究將第肆章分為三大節探討：第一節介紹系統概念用來描述高階主管知識系統的整體運作過程，及大體上系統包括的運作模組，如何使高階主管滿足心智需求；第二節介紹本研究的系統設計，此節探討系統架構圖為結合活動理論與本體論分析高階主管的心智流程及企業營運相關知識，最後運用 JAVA 建構一高階主管知識系統，及系統的功能流程歸於此節一併介紹；；第三節則是介紹系統建置與功能說明。

一、系統概念分析

為了讓高階主管能與所需的企業功能知識相互溝通，本研究須建立一知識系統以滿足高階主管的心智流程。本研究大體上將整個系統大概分為三個運作模組，此三個運作模組，是為了因應高階主管的心智需求所規劃的，一般來說可視為系統建置最理想的情況，因為能根據高階主管的所有需求進行讀取及提供知識內容。第一層模組為需求模組，其主要為接收使用者需求，第二層模組為處理模組，包括四個功能，分別為辨識使用者需求、搜尋使用者所需的資料、彙整及處理所搜尋而得的資料及將彙整過的資料以圖形的方式呈現，第三層模組為資料模組，主要為提供知識內容至處理模組以利運作，如圖 4.1.1 所示：

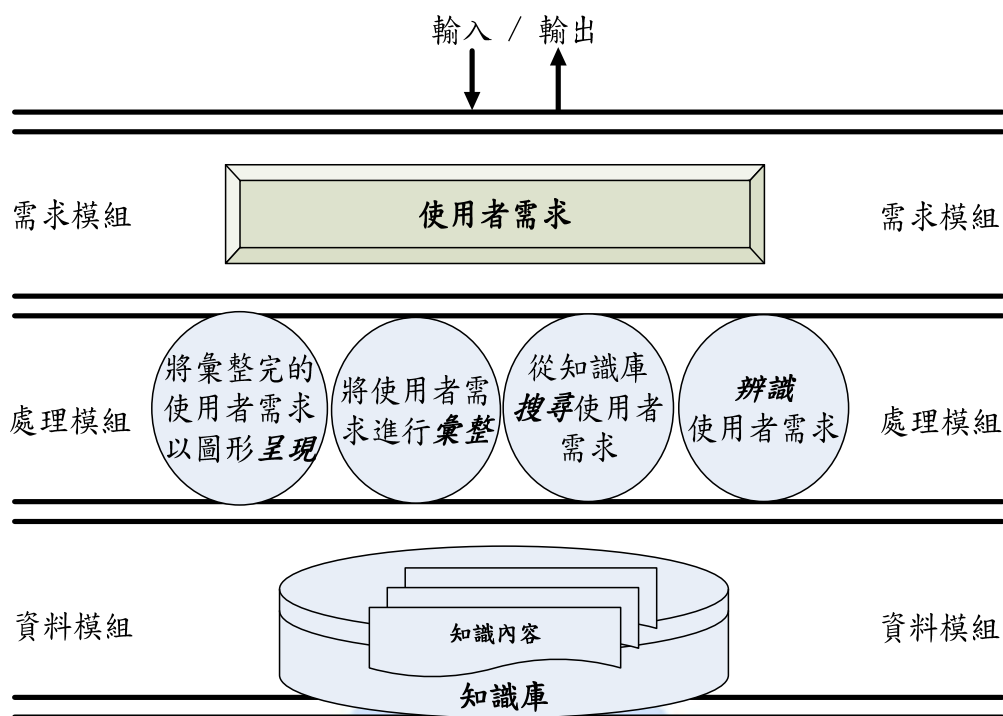


圖 4.1.1 系統概念圖

(一) 需求模組：

需求模組為接收使用者需求，使用者需求為本研究將企業所需的功能知識所分析的知識面。當使用者欲瞭解任何知識面，需求模組則會進行接收使用者需求，接收需求的同時，會連結此需求所位於的知識庫，並將使用者所選擇的知識需求予以呈現。若同時選擇多個知識面，也能根據需求模組進行讀取及比較出其相關的知識內容。使用者可利用此模組重新選擇需求，再重新進行讀取及比較，直到知識需求完全符合使用者的心智需求。

(二) 處理模組：

處理模組可分為四個功能，分別為「辨識」、「搜尋」、「彙整」、「呈現」。

「辨識」功能主要為使用者需求進行辨識，所謂的辨識是將使用者需求進行再一次的辨識，辨識需求模組所接收的使用者需求是否有誤，所謂有誤的情況，代表著使用者需求選擇了重覆的知識。若辨識確認無誤後，辨識功能才能將需求進行下一功能：搜尋功能。「搜尋」功能接收到辨識功能的指令之後，往資料模組進行搜尋使用者需求的動作，所謂的搜尋使用者需求的意思是表示當知識面的需求被辨識無誤後，搜尋功能則會至資料模組找出知識面及其相關的資料，再將搜尋的知識與其相關知識內容傳至彙整功能。「彙整」功能則是根據知識與其相關知識內容，進行推論及彙整後所表達的知識內容則是具有關係的知識，能透過這些關係得知知識與其他知識之間的關係。經過彙整的知識內容，透過「呈現」功能進行表達的方式，讓此功能呈現出最易明瞭的圖形給予使用者。在此，本系統的表達方式及知識來源，理想上可以做到不同的呈現。於表達方式而言，可透過圓餅圖表示各區域銷售量的分佈、各區域的氣候分佈、各區域製造廠房的生產量分佈等；曲線圖表示特定產品、兩種產品、生產、利潤、原物料、成本、銷售在某一年度或某一月份的相互比較；長條圖表示各年度的生產、原物料、利潤、產品分析。於知識來源方面，使用者可根據不同的知識層面所做出來的表達方式進行比較，例如：生產知識面所提供的生產分析與成本知識面所提供的原物料分析一起比較，亦可同時比較多個具有關係的知識面所分析的圖，如行銷知識面的銷售分析與成本知識面的產品分析及客戶知識面的客戶購買情況相互比較，以利高階主管找出其關係點。透過此關係點，能讓高階主管順利的將公司所須改進的地方加以調整，以增加公司的效率及生產力。

(三) 資料模組：

資料模組亦為一知識庫，存放著高階主管經營企業所需的知識內容，包括了本研究所劃分的成本、生產、產品、銷售、區域、競爭者、客戶、原物料等知識內容。當使用者透過需求模組選取任一知識面或多個知識面，處理模組的辨識功能會立即辨識知識需求是否無誤，如果確定符合，搜尋功能則從資料模組裡搜尋知識與其相關知識內容，並傳至彙整功能。因此，資料模組最主要為提供所有知識內容給予處理模組，讓處理模組的四個功能都能順利的將使用者需求進行處理，並讓模組之間相互溝通運作執行以利呈現給使用者。

二、系統設計

本研究除了將大體上系統的運作概念予以描繪之外，也將根據過往高階主管資訊系統所面臨的問題在於資訊不夠集中及知識彼此之間的關係無法有效瞭解，因此本研究採用本體論與活動理論將領域知識本體論進行分析，再透過 Java 建置一溝通介面，此系統將可改善知識庫內容於知識表達、知識分享等方面的效能；並根據圖 4.1.1 的概念將本系統所應俱備的模組及功能加以衍變為本研究的系統架構圖。本研究運用 Java 建構一高階主管知識系統，整合建置於 Protégé 平台上的領域本體論，並利用 Java 的應用程式介面 API (Application Programming Interface, API) 設計出一個能有效的與使用者互相溝通的使用者介面，如圖 4.1.2，將分為兩個部份，一為理論，包括本體論及活動理論，二為系統，包括 OWL 本體論、運作機制與使用者介面。

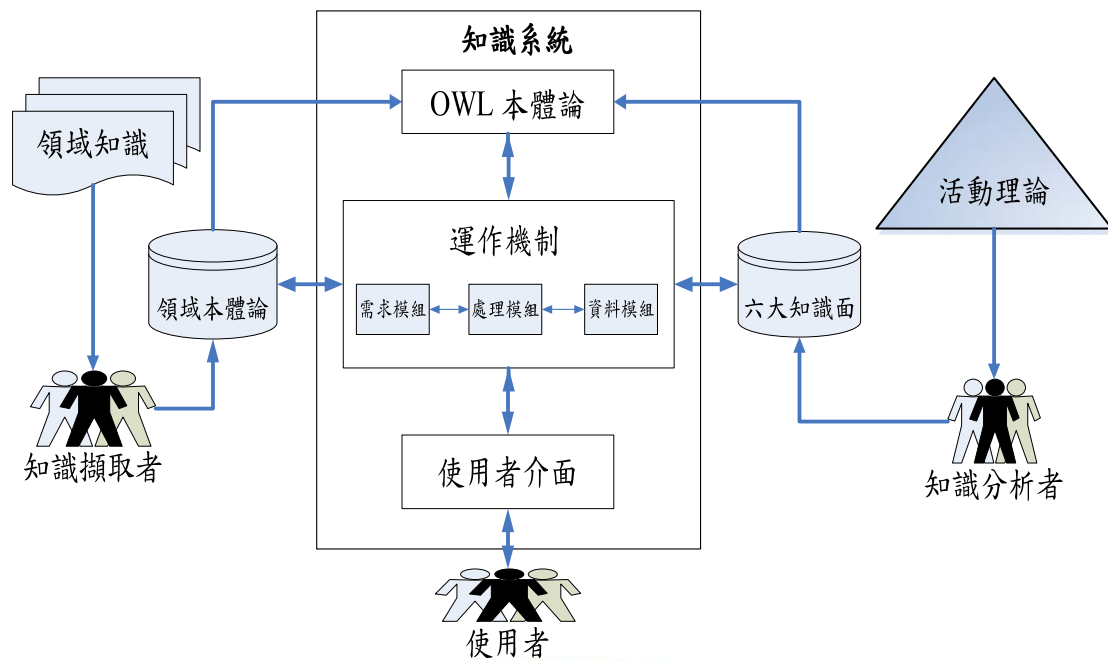


圖 4.1.2 系統架構圖

(一)理論

1. 本體論

本研究運用本體論的觀點分析企業營運所需知識，透過訪談擷取公司各部門的負責人員所得知的知識內容，包括企業功能知識，及其各種知識概念與概念彼此之間的關係。

2. 活動理論：

基於活動理論的基礎，從高階主管的角度分析所得到的知識需求及認知層次架構，可將高階主管的心智流程模式深入的表達，再結合本體論所得知的企業功能知識，即可將高階主管的知識需求予以結合成六大知識面，該六大知識面可提供給高階主管在面臨企業經營時能順利的解決問題，並做出適當的決策。

(二)系統

1. OWL 本體論

結合本體論與活動理論的知識內容後，本研究即可將其知識內容建置為一 OWL 領域知識本體論，此領域本體論亦為一知識庫，該知識庫可提供高階主管存取他所需的知識層面。因此，此一 OWL 本體論的角色是扮演著與系統溝通的橋樑。

2. 運作機制

此運作機制即為本系統所規劃的三個運作模組，透過需求模組接收使用者需求，接著處理模組的四個功能將使用者需求進行辨識、搜尋、彙整及呈現，最後為資料模組配合著需求模組及處理模組，適時的給予資料。因此運作模組在本研究的知識系統中扮演著極為重要的角色，此機制能將系統呈現彈性的功能模式，且唯有透過此運作模組，使用者才能與系統做出良好的溝通。

3. 使用者介面

本研究運用 JAVA 機制來呈現出方便於高階主管與系統有效的溝通介面，這個介面的呈現在於如何將高階主管的心智需求予以瞭解，因此介面的呈現將分為三個部份：一為知識功能層面的提供、二為相關知識內容的呈現、三為知識分析比較。這三個部份皆根據本研究一開始訪談所擷取的高階主管知識所規劃而得知。唯有瞭解高階主管的想法，才能根據他所要的知識層面予以提供，並進而給予適當的知識內容，最後從給予的知識內容以圖形的方法呈現。

本研究根據系統運作的系統概念圖及系統架構圖，將活動理論與本體論分析得知的高階主管知識需求、認知層次架構與企業功能知識，運用系統功能流程圖

將整個系統所應擁有的知識及功能予以分析，說明如下：

(一)系統功能流程

系統功能流程則是將此系統能提供高階主管什麼樣的功能為訴求，這些功能即為系統概念圖裡的處理模組所提到的四大功能：辨識、搜尋、彙整、呈現。有了此四大功能，再結合活動理論所分析得到的高階主管心智流程及工作責任認知層次架構，並運用本體論所建立的高階主管領域知識，及知識層次架構的六大知識面加以運用，透過此二者理論的輔助，能使高階主管方便明瞭企業的營運情形及滿足高階主管所揹負的三大工作責任，最後讓高階主管能適時且適當的做出決策，進而幫助企業解決營運時所面臨的任何問題，如圖 4.1.3 所示：

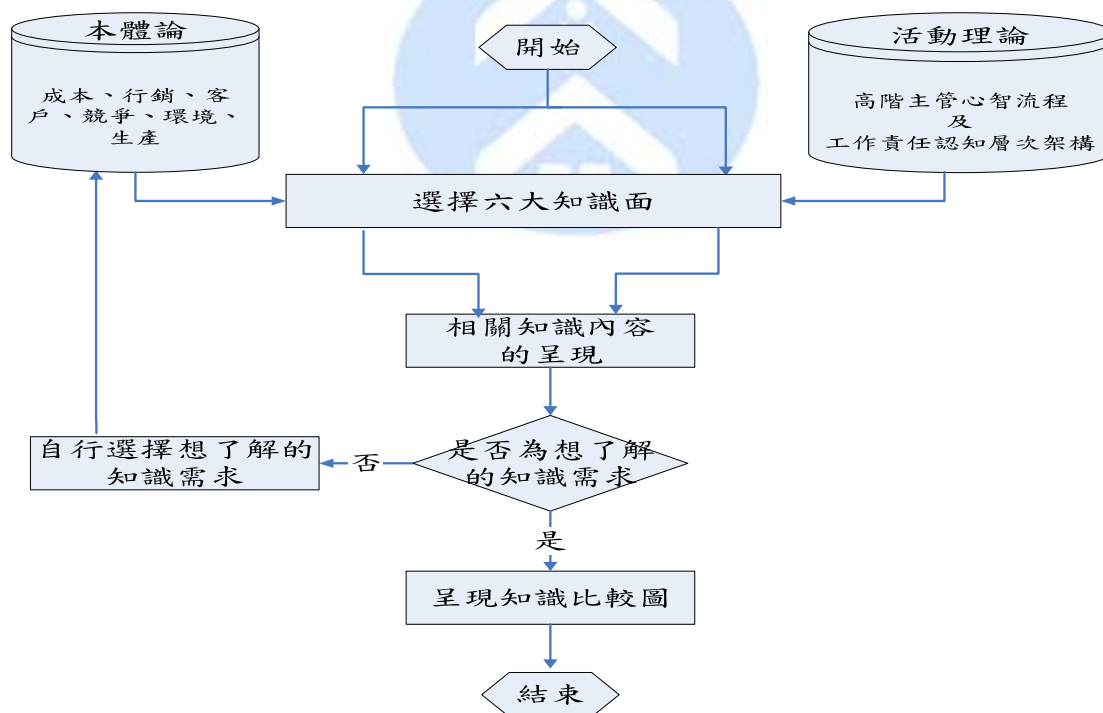


圖 4.1.3 系統功能流程圖

三、系統建置與功能說明

(一) 系統建置

本系統的建置是使用 Protégé 為建構系統的平台，儲存系統所需的 OWL 本體論語言。Protégé 為美國史丹佛大學 SMI (Stanford Medical Informatics) 中心以 Java 所研究開發的知識庫自由軟體，是目前最廣泛使用的平台之一。Protégé Core System 輔助知識工作者使用其 OWL 建構本體論。高階主管透過以 Java 撰寫的高階主管知識系統，並且還能透過系統根據所選擇的知識類別的各種相關資訊存入知識庫中，如下圖 (圖 4.2.1)。左方框框為高階主管知識領域本體論所建立的類別階層；中間底下的框框為根據編輯的類別進行屬性設定及限制語法，此舉能讓本體論的關係更為密切，推論機制也會更加強大。



圖 4.2.1 使用 Protégé 平台所建構的高階主管知識本體論

(二) 系統功能說明

此節將敘述本研究的系統功能，以系統所劃分的三大運作模組及其中所運作的功能做說明，功能將分別為辨識、搜尋、彙整、呈現。本系統主要分為三個操作畫面：

1. 主畫面

主畫面代表著系統的需求模組，為接收使用者需求。其中運作的功能除了接收使用者需求，也將為使用者需求進行辨識。當高階主管選擇欲瞭解的知識面，此辨識功能將會根據高階主管所選擇的知識面進行判斷及辨識，若判斷及辨識此知識需求無誤且符合高階主管需求，即能將此需求透過處理模組之搜尋、彙整功能來執行，最後進行呈現。

2. 知識需求的內容

此階段為本系統處理模組中的搜尋、彙整功能，本系統將知識需求進行辨識後，接著至資料模組搜尋此知識需求，最後將其知識需求予以推論出相關知識內容，最後加以彙整呈現。

3. 知識類別的圖表分析

圖表分析為處理模組的呈現功能，是經過搜尋、彙整功能所得知的知識內容，透過此圖表分析將知識內容以圖形的表達方式呈現，以利於進行比較。

就整個系統的運作而言，本研究所規劃的三個運作模組，各自扮演著在存取知識需求的過程中相當重要的角色，且此三個運作模組，更讓本系統與以往的 EIS 大不相同。因為需求模組、處理模組、資料模組皆能針對不同的使用者需求進行

判斷及推論，反覆的找出符合使用者需求的知識，讓整個系統在知識需求的呈現上為一彈性的運作模式，此舉不僅提升了存取知識的深度及廣度，也讓本系統解決了以往 EIS 無法滿足高階主管心智需求的問題。

以下將以一知識面解釋各功能如何溝通運作：

(1)系統主畫面



圖 4.2.2 高階主管知識系統首頁

如上圖(圖 4.2.2)所示，本研究將此系統分為六大知識面可供高階主管存取，這六大知識面是經由訪談公司所擷取的知識需求，此知識需求皆為高階主管經營企

業時所需要的知識層面。因此，本系統提供高階主管自行選擇欲瞭解的知識面，透過所選擇的知識面，系統會先接收需求及連結系統所建置的 OWL 本體論，之後再進行辨識功能所運作的辨識及比對。

(2)相關知識內容分析



圖 4.2.3 知識需求與 OWL 本體論的連結

如圖 4.2.3 所示，當使用者選擇欲瞭解的知識面之後，會先與本系統所建置的 OWL 領域知識本體論進行連結及讀取，當系統進行讀取之後，將可看到使用者所

選取的知識面會根據主要知識及次要知識做出分類，讓使用者瞭解此知識面所提供的知識內容之重要程度與否。

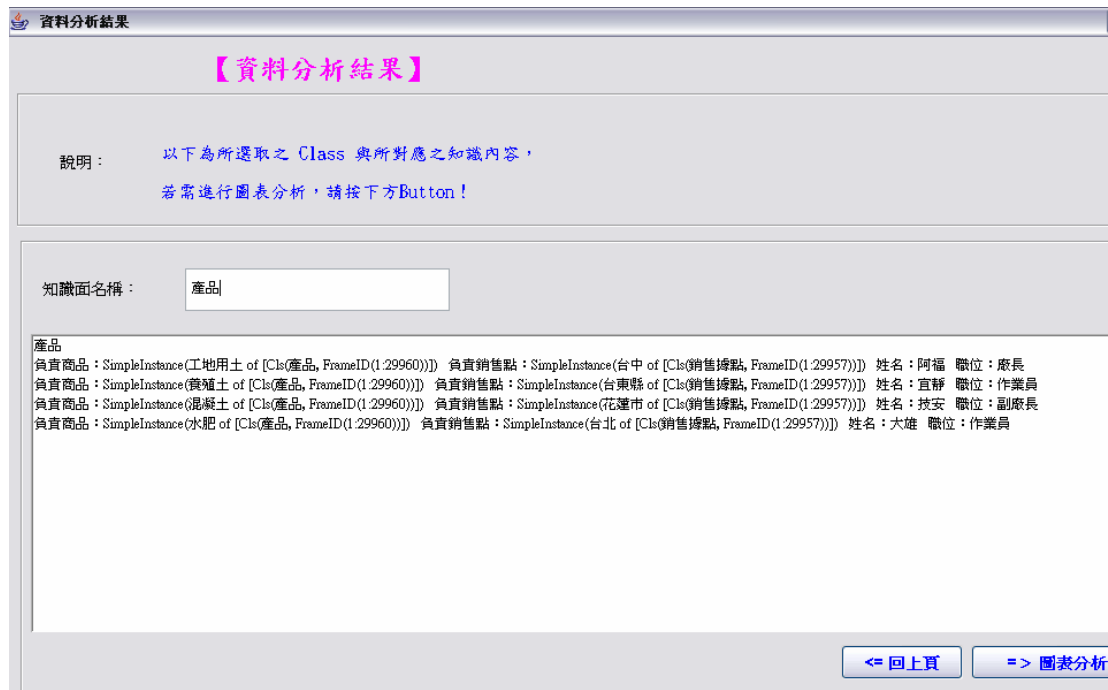


圖 4.2.4 知識需求相關的知識內容圖

當使用者選擇主要知識或是次要知識的同時，本系統可透過使用者所選取的知識需求進行推論的動作，將知識需求與其他知識需求的關係於 OWL 進行推論及呈現，如圖 4.2.4。因此，所呈現的結果，與傳統 EIS 所顯現的功能最大的不同在於，本系統可根據使用者的需求進行判斷及推論出相關的知識，也意謂根據使用者的需求進行推論，代表著本系統的功能模式是屬於彈性的模式，與以往 EIS 固定的功能模式大不相同，因為以往 EIS 的固定功能模式，無法因應使用者的需求變動，只能將 EIS 內定的功能予以提供，並不能反應出使用者的心智需求。

(3)知識類別分析圖

透過彙整功能所呈現的相關知識內容除了與以往 EIS 固定功能模式不同之外，本系統最後一個知識類別分析圖，更是根據相關的知識內容將使用者的需求以圖形的表示方法呈現出知識需求，雖然傳統的 EIS 也提供圖形的表達，但是本系統能進行比較的分析，可根據使用者所選擇的知識需求，進行年度的分析比較，一樣是屬於彈性的功能模式；若所呈現的知識相關內容與使用者的需求不符，使用者也能再一次自行選擇欲瞭解的知識層面。

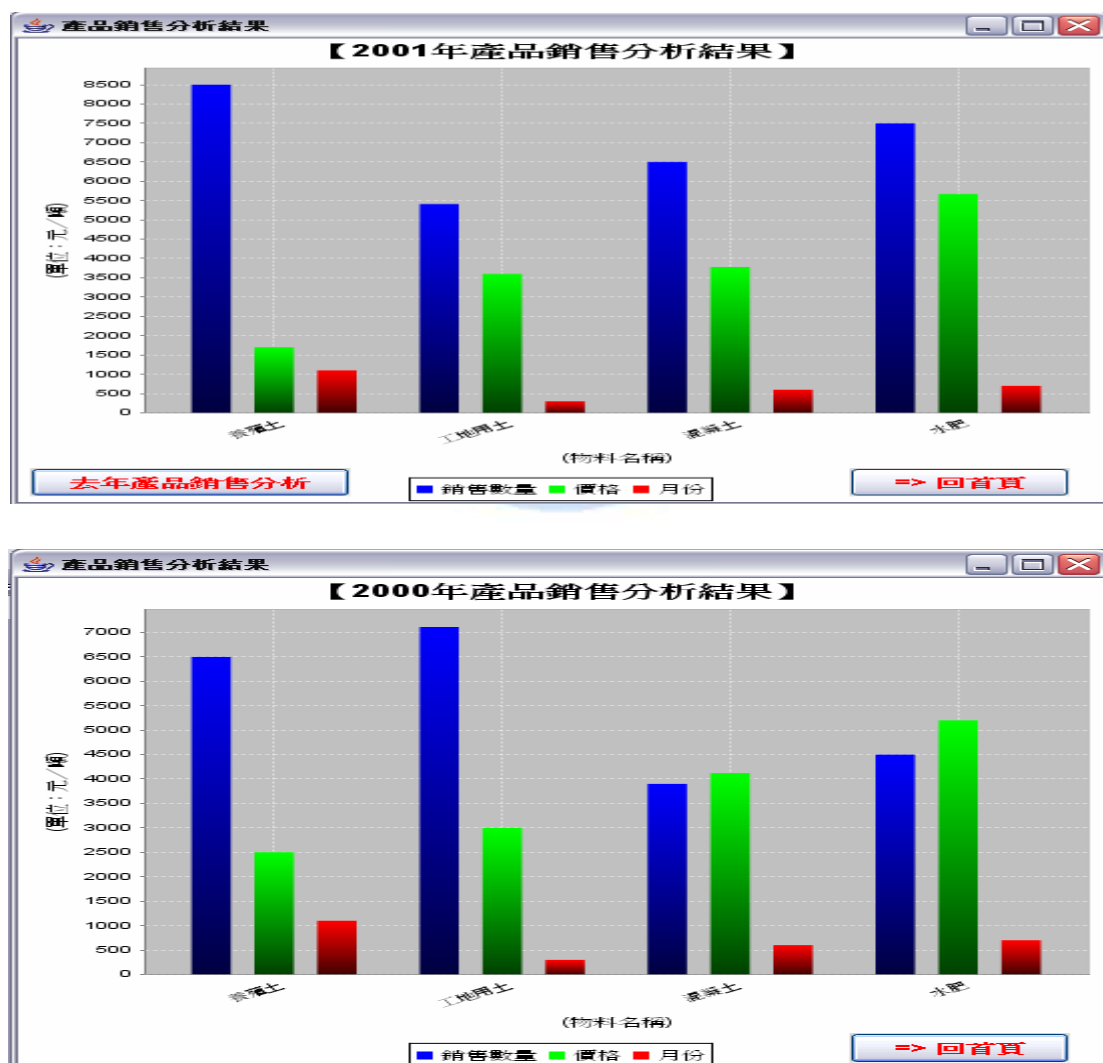


圖 4.2.5 知識類別分析圖

伍、結論與未來研究方向

有鑑於高階主管的責任及所需知識是屬於不確定性、變動性，以及高階主管所面臨的工作環境是如此的龐大及複雜，但是 EIS 只著重在固定功能的提供、特定問題的解決分析，若高階主管需求變動，EIS 無法針對變動的需求做出即時的因應，導致無法滿足高階主管於知識需求。EIS 亦無法完全即時提供高階主管需要的資料及適時的滿足高階主管的決策知識需求，使得在高階主管知識內容的表達上，讓高階主管不易瞭解 EIS 所想要提供的功能為何，系統設計者也無法幫助高階主管來給予適當的資訊。因此本研究應用活動理論將高階主管的心智流程予以分析後，利用活動階層性架構分析高階主管應負的工作責任，之後再經由本體論分析出高階主管企業功能知識間彼此的關係，最後運用 JAVA 建構一高階主管知識系統。

一、結論

本研究透過應用活動理論與本體論建置一高階主管知識系統，於建置系統前，本研究分析高階主管所具備的知識內容、認知層次架構及企業功能知識相互之間的關係，在此運用 Leontiev (1981)學者所提出的活動階層性架構的概念，此概念最主要在於將人們內在認知的過程轉變為外在執行的顯現，在這樣轉變的過程中，即可發現人們的內心需求是如何透過外在活動來表示。因此，活動階層性架構的概念將可以有效的解決過去 EIS 所無法做到的部份，亦即 EIS 只能提供固定的功能模式，無法因應高階主管知識需求的變動性及無法瞭解高階主管心智需求。故本研究藉由運用活動理論分析出高階主管的決策知識心智流程，此舉不僅

能改善 EIS 提供知識時只有固定的功能模式，且知識不夠深入與太過分散的缺點，又能幫助高階主管在面臨企業營運之問題時，能有效的運用本系統所劃分的三個運作模組，透過三個模組的相互溝通及功能的運作，將可以根據高階主管所需的知識面給予適當的表達方式以呈現給高階主管，亦會讓高階主管自行選擇所欲瞭解的知識面，進而比較不同的知識面，以提供高階主管何者知識為直接的主要知識及聯想的次要知識，以利適時的做出適當的決策，最後滿足高階主管所揹負的三項責任：追求利潤、提升競爭能力、永續經營。因此本研究的主要貢獻如下：

(一) 應用活動理論分析高階主管決策知識心智流程

本研究運用活動理論之活動階層性架構來分析高階主管的心智流程、心智需求與高階主管經營企業所需揹負的三大工作責任與所需的企業功能知識，此舉可有效解決以往 EIS 不瞭解高階主管的心智需求，導致無法提供高階主管正確且適時的知識。因此，運用活動理論分析所得知的高階主管決策知識心智流程將能有效解決這樣的問題。

(二) 高階主管系統概念架構

本研究於建置系統前，須規劃分析出本系統該有的概念架構，才能順利的將高階主管所想要的知識內容予以呈現。因此，本研究將系統的概念架構分為三個運作模組，一為需求模組、二為處理模組、三為資料模組。本研究將系統分析為三個運作模組的用意是為了能有效接收高階主管的知識需求，到如何去搜尋及彙整處理知識需求的過程，以及最後所呈現的一個表達方式，皆能根據使用者的需求進行判斷及推論，進而符合高階主管的心智流程的運作過程。此一運作過程，更能有效改善以往 EIS 所提供的固定功能模式，因為本系統為高階主管設計的三

個模組，能針對高階主管直接的主要知識及聯想的次要知識做出判斷，進而推論其知識需求以滿足高階主管的工作責任。

(三) 應用 OWL 與 JAVA 建構一高階主管知識系統

本研究運用本體論分析企業功能知識，透過 OWL 建置為高階主管可存取的領域知識本體論，當高階主管面臨企業決策問題時，即可快速的透過此本體論瞭解高階主管經營企業直接需要的主要知識及聯想到的次要知識，透過主要知識及次要知識的分類，OWL 本體論與運作機制才能有效因應高階主管的需求變動。為了讓系統與高階主管能有一互相溝通的機制，本研究運用 JAVA 建立一溝通介面，可讓高階主管透過此系統與企業功能知識相互溝通，且能依照高階主管的心智流程得到想要的知識，進而比較其相關知識內容。

二、未來研究方向

由於每間公司高階主管的企業營運知識會因公司的特性或是性質而產生不同的知識，所以就此方面而言，高階主管的知識是相當廣泛的，本研究僅就一間製造公司分析高階主管所需的企業營運知識內容。因此，針對本研究未來可繼續執行的目標及重點，歸納出三個主要的方向：領域本體論的延伸、系統績效的衡量及系統未來呈現的方式，如下：

(一) 領域本體論的延伸

本研究的領域本體論是根據高階主管所需的知識需求去建立出功能給予高階主管使用，故僅站在高階主管的角度觀看所考慮到的知識需求層面，因為高階主

管所需揹負的工作責任是一間公司是否能繼續生存發展的最大因素。因此在高階主管知識內容部份，未來可將一般主管或是針對其他知識面進而擴展，加入於本體論中，以建構出更完整的領域本體論供所有主管級相關人員使用。

(二)系統績效的衡量

身為一高階主管，要經營一家企業是一項相當困難的工作，且其涵蓋範圍十分廣泛，本研究僅以某間公司為主，替高階主管進行企業知識需求分析。然而，公司資料為商業機密，難以取得實際資料進行系統之衡量，無法使得系統更加的完善。因此，若能取得實際資料進行評估，相信本研究所建置的系統在呈現績效時更能精確且符合公司的需求。此外，本研究是藉由一家公司的訪談來擷取其高階主管知識內容，因此所建立的高階主管決策知識心智流程是屬於一般性質，系統也是根據此心智流程所建構，所以系統的功能運作並不能代表所有高階主管的心智需求。

(三)系統未來呈現的方式

本研究所建置的系統，目前只能呈現一種圖示的方法，此方法為長條圖的呈現方式。但是，高階主管若想根據某一種主力產品或主力分析事項時，本系統無法將某特定產品以曲線圖呈現某一年度或某一月份的表達方式或是某區域產品銷售量，也無法以圓餅圖的方式呈現出各區域所佔的比重，因此系統未來呈現的方式，可以運用更多圖示的 Plug-in 將系統所分析出來的結果以更多元化的方式呈現，例如曲線圖、圓餅圖等。

陸、參考文獻

一、中文部份

1. 阮明淑，溫達茂(民 91)。ontology 應用於知識組織之初探。佛教圖書館館訊，12 月第 32 期。
2. 葉文權(民 94)。應用本體論建構財務報表分析專家系統。國立高雄第一科技大學資訊管理所碩士論文。
3. 張成儀(民 95)。應用本體論建構短期授信專家系統。國立高雄第一科技大學資訊管理所碩士論文。

二、英文部份

1. Aguilar, F.J. (1967). Scanning the Business Environment, *Macmillan, New York*.
2. Anongnart, S. (1999). Understanding Executive Information Systems Implementation: an Empirical Study of EIS Success Factors, *IEEE Proceedings of the 32nd Hawaii International Conference on System Sciences*.
3. Ashock, A. and Beck, C. (2007). Using Activity Theory to Develop a Design Framework for Rural Development, *ACM, CHI'07, San Jose, California, USA*.
4. Boer, N.I., VanBaalen, P.J. and Kumar, K. (2002). An Activity Theory Approach for Studying the Situatedness of Knowledge Sharing, *IEEE Proceedings of the 35th Hawaii International Conference on System Sciences*.
5. Bunge, M. (1977). Ontology I: The Furniture of the World, *Treatise on Basic Philosophy, 3, D Reidel*.

6. Castro, J., Neto, G.C., Gomes, A.S. and Sampaio, S. (2005). Integrating Activity Theory and Organizational Modeling for Context of Use Analysis, *ACM Cuernavaca, Mexico, CLIHC'05*.
7. Chandrasekaran, B., Josephson, J.R. and Benjamins, V.R. (1999). What Are Ontologies, and Why Do We Need Them? , *IEEE Intelligent Systems*, 14(1), 20-26.
8. Corcho, O., Fernandez, L.M. and Gomez, P.A. (2003). Methodologies, tools and languages for building ontologies. Where is their meeting point? *Elsevier Science Data & Knowledge Engineering* 46, 41–64.
9. Daft, R. and Lengel, R.(1986). Organizational Information Requirements, Media Richness and Structural Design, *Management Science* (32:4), 554-571.
10. Engeström, Y. (1987). Learning by expanding: an activity theoretical approach to developmental research, *Helsinki:Orenta-Konsultit*.
11. Engeström, Y. and Kuutti, K. (2006). Activity Theory. *Elsevier*.
12. Fensel, D. (2000). The semantic Web and its languages, *IEEE INTELLIGENT SYSTEMS*.
13. Fernandez, M. L. (1999), Overview of Methodologies For Building Ontologies, *Proceedings of the Workshop on Ontologies and Problem-Solving Methods, International Joint Conference on AI (IJCAI-99), Sweden*.
14. Fernandez, M.L., Gomez-Perez, A. and Juristo, N. (1997). METHON- TOLOGY: From Ontological Art Towards Ontological Engineering. *In Proc. of AAAI97 Spring Symposium Series, Workshop on Ontological Engineering, AAAI Press, 33-40*.

15. Freitas, M.R. and Byrne, E. (2006). Activity Theory as an Analytical Tool: A Case Study of IS Development for an Anti-Retroviral Treatment Clinic in South Africa, *ACM Proceedings of SAICSIT'06*, 90 –99.
16. Grosso, W.E., Eriksson, H., Ferguson, R. W., Gennari, J. H., Tu, S.W. and Musen, M.A. (1999). Knowledge Modeling at the Millennium: The Design and Evolution of Protege-2000, *SMI Technical Report, SMI-1999-0801*.
17. Gruber, T.R. (1993). A Translation Approach to Portable Ontology Specifications, *Knowledge Acquisition*, 5, 199–220.
18. Gruninger, M. and Fox, M. S. (1995). Methodology for the Design and Evaluation of Ontologies, *Department of Industrial Engineering University of Toronto, Canada, M S A*.
19. Hasan, H. (2002). An Activity-based Model of Collective Knowledge, *IEEE Proceedings of the 36th Hawaii International Conference on System Sciences*.
20. Hasan, H. and Gould, E. (2001). Support for the sense-making activity of managers, *Decision Support Systems* 31, 71-86.
21. Horridge, M., Knublauch, H., Rector, A., Stevens, R. and Wroe, C. (2004). A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools, *Stanford University and The University Of Manchester*.
22. Karp, P.D. (1993). The Design Space of Frame Knowledge Representation Systems, *Artificial Intelligence Center, SRI International, CA 94025*.
23. Kim, H. M.(2002). XMLhoo!: A Prototype Application for Intelligent Query of XML Documents using Domain-Specific Ontologies, *Proceedings of the 35th Hawaii International Conference on Systems Science (HICSS-35 '02), USA*.

24. Kotter, J.P. (1982). What Effective General Managers Really Do, *Harvard Business Review* (60:6), 160.
25. Kuutti, K. (1996). Activity Theory as a potential framework for Human-computer interaction research, B.A. Nardi, (ed.), *context and consciousness: Activity Theory and Human-Computer Interaction*. Cambridge, MA: MIT press, 17-44.
26. Leontiev, A. N. (1981). Problems of the Development of Mind, *Moscow: Progress*.
27. Leung, K.S. and Wong, M.H. (1990). An Expert-System Shell Using Structured Knowledge – An Object-Oriented Approach, *IEEE COMPUTER*, 38-47.
28. McClatchy, W. (1990). Meadows the CEO, *Information Week*, 34
29. McGuinness, D.L. and Harmelen, F.V. (2004). *OWL Web Ontology Language Overview* W3C Recommendation, <http://www.w3.org/TR/2004/REC-owl-features-20040210/>
30. Musen, M. A., Noy, N. F. and Ferguson, R. W. (2003). The knowledge model of Protégé-2000: Combining interoperability and flexibility, *Proceedings of the 2thInternational Conference on Knowledge Engineering and Knowledge Management, France*.
31. Noy, N.F. and McGuinness, D.L., *Ontology Development 101: A Guide to Creating Your First Ontology*, Stanford University, Stanford, CA, 94305.
32. Noy, N.F., Sintek, M., Decker, S., Crubézy, M., Ferguson, R.W. and Musen, M. A. (2001). Creating Semantic Web Contents with Protégé-2000, *IEEE INTELLIGENT SYSTEMS*.
33. Paller, A. and Laska, R. (1990). *The EIS Book*, Dow Jones-Irwin, Homewood, IL.
34. Peters, S. and Shrobe, H. E. (2003). Using Semantic Networks for Knowledge

- Representation in an Intelligent Environment, *IEEE COMPUTER SOCIETY*, 1-7.
35. Rainer Jr., R.K. and Watson, H. J. (1995). What does it take for successful executive information systems? *Decision Support Systems 14*, 147-156.
 36. Rockart, J.F. and Treacy, M.E. (1982). The CEO goes online, *Harvard Business Review*.
 37. Salmeron, J. L. (2002). EIS data: findings from an evolutionary study, *The Journal of Systems and Software 64*, 111–114.
 38. Simon, H.A. (1957). Administrative Behavior, *Macmillan, New York*.
 39. Stecklow, S. (1989). The New Executive Information Systems, *Lotus*, 51-55.
 40. Sugumaran, V. and Storey, V.C. (2002). Ontologies for conceptual modeling: their creation, use, and management, *Elsevier Science Data & Knowledge Engineering 42*, 251–271.
 41. Turban, E. and Waston, H. J. (1987). Integrating Expert Systems, Executive Information Systems, and Decision Support Systems, *DSS 89 Transactions, The Institute of Management Sciences, Providence, RI*, 74-82.
 42. Uden, L. and Willis, N. (2001). Designing User Interfaces using Activity Theory. *IEEE Proceedings of the 34th Hawaii International Conference on System Sciences*.
 43. Uschold, M. and King, M. (1995). Towards a Methodology for Building Ontologies, *Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on AI (IJCAI-95), Canada*.
 44. Victor K. and Bonnie, A.N. (1997). Activity Theory: Basic Concepts and Applications, *TUTORIALS CHI 97*, 22-27.
 45. Vygotsky, L.S. (1978). Mind and Society, *Harvard University Press, Cambridge*,

46. Walters, B.A., Jiang, J.J. and Klein, G. (2003). Strategic Information and Strategic Decision Making:the EIS/CEO Interface in Smaller Manufacturing Companies, *ELSEVIER, Information & Management 40* , 487–495.
47. Wang, X. H., Zhang, D.Q., Gul, T. and Pung, H.K. (2004). Ontology Based Context Modeling and Reasoning using OWL, *IEEE Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*.
48. Watson, H. J., Rainer Jr., R.K. and Koh, C. E. (1991). Executive Information Systems: A framework for development and a survey of current practices, *MIS Quarterly*, 15(1), 13-30.
49. Watson, H.J. and Frolick, M.N. (1993). Determining Information Requirements for an EIS, *MIS Quarterly*, 17(3), 255-269.
50. Watson, H.J., Watson, R.T., Singh, S. and Holmes, D. (1995). Development practices of executive information systems: finding of a field study, *Decision Support Systems 14*, 171-184.
51. Watson, H.J., Houdeshel, G. and Rainer Jr., R.K.(1997). Building Executive Information Systems and Other Decision Support Applications, *John Wiley & Sons, New York*.
52. Yang, Y.K. (1990). Behind The Inheritance Relations In a Semantic Network, *Southeastcon '90. Proceedings, IEEE, 1*, 289-296.
53. Young, D. and Watson, H.J. (1995). Determinates of EIS acceptance, *Information & Management 29*, 153-164.

附錄一、介面的呈現

一、主畫面

```
package nkfust;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import com.borland.jbcl.layout.*;
import java.io.*;
import java.util.*;
import java.awt.Dimension;
import javax.swing.BorderFactory;
import java.awt.Font;
public class EIS extends JFrame {
    JPanel contentPane;
    JPanel jPanel2 = new JPanel();
    XYLayout xYLayout1 = new XYLayout();
    XYLayout xYLayout2 = new XYLayout();
    JLabel jLabel1 = new JLabel();
    JLabel jLabel2 = new JLabel();
    JLabel jLabel3 = new JLabel();
    JLabel jLabel4 = new JLabel();
    JLabel jLabel5 = new JLabel();
    JLabel jLabel6 = new JLabel();
    JLabel jLabel7 = new JLabel();
    JLabel jLabel8 = new JLabel();
    JLabel jLabel9 = new JLabel();
    JButton jButton1 = new JButton();
    JButton jButton2 = new JButton();
    JButton jButton3 = new JButton();
    JButton jButton5 = new JButton();
    JButton jButton6 = new JButton();
    JButton jButton7 = new JButton();
```

```

JButton jButton8 = new JButton();
JTextField jTextField1 = new JTextField();
JList jList1;
JList jList2;
private DefaultListModel listModel1;
private DefaultListModel listModel2;
OWLClass OC;
OWLInstance OI;
OWLProperty OP;
Vector vctPro1 = new Vector();
Vector vctPro2 = new Vector();
String strEvent, strName;
public static int screenwidth,screenheight;
EIS jEIS;
public static EIS frame;
JLabel jLabel10 = new JLabel();
JButton jButton9 = new JButton();
JButton jButton10 = new JButton();
JLabel jLabel11 = new JLabel();
public EIS() {
    enableEvents(AWTEvent.WINDOW_EVENT_MASK);
    try {
        jbInit();
    }
    catch(Exception e) {
        e.printStackTrace();
    }
    addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            dispose();
            System.exit(0);
        }
    });
}

```

```

//Main method
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
    }
    catch(Exception e) {
        e.printStackTrace();
    }
    frame = new EIS();

    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    Dimension frameSize = frame.getSize();
    if (frameSize.height > screenSize.height) {
        frameSize.height = screenSize.height;
    }
    if (frameSize.width > screenSize.width) {
        frameSize.width = screenSize.width;
    }
    screenwidth = (screenSize.width - frameSize.width) / 3;
    screenheight = (screenSize.height - frameSize.height) / 3;
    frame.setLocation((screenSize.width - frameSize.width) / 3, (screenSize.height
- frameSize.height) / 3);
    frame.setVisible(true);
}

private void jbInit() throws Exception {
    contentPane = (JPanel) this.getContentPane();
    contentPane.setLayout(xYLayout2);
    this.setSize(new Dimension(709, 600));
    this.setTitle("高階主管知識系統");
    Vector v1 = new Vector();
    listModel1 = new DefaultListModel();
    listModel2 = new DefaultListModel();
    jList1 = new JList(listModel1);
}

```

```
jList2 = new JList(listModel2);
jList1.setVisible(true);
jList1.setAlignmentY((float) 0.5);
jList2.setVisible(true);
jList2.setAlignmentX((float) 0.5);
jList2.setAutoscrolls(true);
jPanel2.setLayout(xYLayout1);
jLabel1.setBorder(BorderFactory.createEtchedBorder());
jButton1.setFont(new java.awt.Font("Dialog", 0, 14));
jButton1.setHorizontalAlignment(SwingConstants.CENTER);
jButton1.setSelected(false);
jButton1.setSelectedIcon(null);
jButton1.setText("成本知識面");
jButton2.setFont(new java.awt.Font("Dialog", 0, 14));
jButton2.setText("行銷知識面");
jButton3.setFont(new java.awt.Font("Dialog", 0, 14));
jButton3.setText("客戶知識面");
jButton5.setFont(new java.awt.Font("Dialog", 0, 14));
jButton5.setAlignmentY((float) 0.5);
jButton5.setText("環境知識面");
jButton9.setFont(new java.awt.Font("Dialog", 0, 14));
jButton9.setText("競爭者知識面");
jButton10.setFont(new java.awt.Font("Dialog", 0, 14));
jButton10.setText("生產知識面");
jButton6.setFont(new java.awt.Font("Dialog", 1, 14));
jButton6.setForeground(Color.blue);
jButton6.setText("× 刪除");
jButton7.setFont(new java.awt.Font("Dialog", 1, 14));
jButton7.setForeground(Color.blue);
jButton7.setText("=> 分析");
jButton8.setFont(new java.awt.Font("Dialog", 0, 14));
jButton8.setText(">>");
jLabel2.setFont(new java.awt.Font("標楷體", Font.PLAIN, 16));
jLabel2.setForeground(SystemColor.desktop);
```

```
jLabel2.setBorder(null);
jLabel2.setToolTipText("");
jLabel2.setText("您好，歡迎使用本系統，");
jLabel3.setFont(new java.awt.Font("標楷體", Font.PLAIN, 16));
jLabel3.setForeground(SystemColor.desktop);
jLabel3.setRequestFocusEnabled(true);
jLabel3.setToolTipText("");
jLabel3.setText("此系統最主要是為了要提供高階主管在");
jLabel4.setFont(new java.awt.Font("標楷體", Font.PLAIN, 16));
jLabel4.setForeground(SystemColor.desktop);
jLabel4.setText(" 企業營運相關的知識，");
jLabel5.setFont(new java.awt.Font("標楷體", Font.PLAIN, 30));
jLabel5.setForeground(SystemColor.desktop);
jLabel5.setText("【高階主管知識系統】");
jLabel6.setFont(new java.awt.Font("標楷體", Font.PLAIN, 16));
jLabel6.setForeground(SystemColor.desktop);
jLabel6.setText("若想選其他知識面，請按回首頁重新選擇。");
jLabel7.setFont(new java.awt.Font("標楷體", 0, 14));
jLabel7.setBorder(BorderFactory.createEtchedBorder());
jLabel7.setText("");
jLabel8.setFont(new java.awt.Font("標楷體", 0, 14));
jLabel8.setForeground(SystemColor.activeCaptionText);
jLabel8.setToolTipText("");
jLabel8.setText("請選擇欲了解的知識面：");
jLabel9.setBorder(BorderFactory.createEtchedBorder());
jLabel9.setText("");
jLabel10.setFont(new java.awt.Font("標楷體", 0, 14));
jLabel10.setForeground(SystemColor.activeCaptionText);
jLabel10.setText("使用說明：");
jLabel11.setText("此系統共分為六大知識面！");
jLabel11.setToolTipText("");
jLabel11.setRequestFocusEnabled(true);
jLabel11.setForeground(SystemColor.desktop);
jLabel11.setFont(new java.awt.Font("標楷體", Font.PLAIN, 16));
```

```

contentPane.add(jPanel2, new XYConstraints(0, 0, 0, 0));
contentPane.add(jLabel10, new XYConstraints(15, 98, 83, 34));
contentPane.add(jButton6, new XYConstraints(531, 405, 90, 30));
contentPane.add(jButton7, new XYConstraints(531, 480, 90, 30));
contentPane.add(jButton10, new XYConstraints(494, 205, 117, 30));
contentPane.add(jButton3, new XYConstraints(314, 205, 153, 30));
contentPane.add(jLabel6, new XYConstraints(99, 114, 488, 26));
contentPane.add(jLabel3, new XYConstraints(98, 86, 315, 26));
contentPane.add(jLabel2, new XYConstraints(102, 55, 185, 35));
contentPane.add(jLabel11, new XYConstraints(98, 144, 201, 26));
contentPane.add(jLabel1, new XYConstraints(13, 44, 677, 139));
contentPane.add(jLabel4, new XYConstraints(359, 85, 174, 29));
contentPane.add(jButton9, new XYConstraints(313, 253, 155, 30));
contentPane.add(jButton5, new XYConstraints(494, 252, 117, 30));
contentPane.add(jLabel7, new XYConstraints(12, 187, 677, 144));
contentPane.add(jLabel9, new XYConstraints(12, 334, 676, 228));
contentPane.add(jButton1, new XYConstraints(176, 205, 113, 30));
contentPane.add(jButton2, new XYConstraints(175, 254, 114, 30));
contentPane.add(jLabel8, new XYConstraints(14, 199, 158, 44));
contentPane.add(jList2, new XYConstraints(341, 358, 150, 190));
contentPane.add(jButton8, new XYConstraints(243, 430, 70, 30));
contentPane.add(jList1, new XYConstraints(68, 359, 150, 190));
contentPane.add(jLabel5, new XYConstraints(202, 5, 304, 34));
contentPane.add(jLabel1, new XYConstraints(9, 50, 525, 129));
    jList1.setBorder(BorderFactory.createLoweredBevelBorder());
    jList2.setBorder(BorderFactory.createLoweredBevelBorder());
    JButtonResponser brButtonTrigger = new JButtonResponser();
    jButton1.addActionListener(brButtonTrigger);
    jButton2.addActionListener(brButtonTrigger);
    jButton3.addActionListener(brButtonTrigger);
    jButton5.addActionListener(brButtonTrigger);
    jButton6.addActionListener(brButtonTrigger);
    jButton7.addActionListener(brButtonTrigger);
    jButton8.addActionListener(brButtonTrigger);

```

```

        jButton9.addActionListener(brButtonTrigger);
        jButton10.addActionListener(brButtonTrigger);
    }

protected void processWindowEvent(WindowEvent e) {
    super.processWindowEvent(e);
    if (e.getID() == WindowEvent.WINDOW_CLOSING) {
        System.exit(0);
    }
}

class jButtonResponser implements ActionListener {
public void actionPerformed(ActionEvent e) {
    int intResult;
    if(e.getActionCommand().equals("成本知識面")) {
        OC= new OWLClass();
        try {
            strName = OC.getClassName("成本知識面");
            listModel1.addElement(strName);
            jTextField1.setText(strName);
        }
        catch(Exception ex) {
        }
    }
    else if(e.getActionCommand().equals("行銷知識面")) {
        OC = new OWLClass();
        try {
            strName = OC.getClassName("行銷知識面");
            listModel1.addElement(strName);
            jTextField1.setText(strName);
        }
        catch(Exception ex) {
        }
    }
}
}

```



```

else if(e.getActionCommand().equals("客戶知識面")) {
    OC = new OWLClass();
    try {
        strName = OC.getClassName("客戶知識面");
        listModel1.addElement(strName);
        jTextField1.setText(strName);
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals("環境知識面")) {
    OC = new OWLClass();
    try {
        strName = OC.getClassName("環境知識面");
        listModel1.addElement(strName);
        jTextField1.setText(strName);
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals("環境知識面")) {
    OC = new OWLClass();
    try {
        strName = OC.getClassName("環境知識面");
        listModel1.addElement(strName);
        jTextField1.setText(strName);
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals("生產知識面")) {
    OC = new OWLClass();
    try {
        strName = OC.getClassName("生產知識面");

```

```

        listModel1.addElement(strName);
        jTextField1.setText(strName);
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals("競爭者知識面")) {
    OC = new OWLClass();
    try {
        strName = OC.getClassName("競爭者知識面");
        listModel1.addElement(strName);
        jTextField1.setText(strName);
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals("=> 分析")) {
    try {
        Value Valueframe = new Value();
        Valueframe.setLocation(650,600);
        Valueframe.setVisible(true);

```

```

Valueframe.frameUIInit(jTextField1.getText(),jList2.getModel().toString());
    }
    catch(Exception ex) {
    }
}

```

```

else if(e.getActionCommand().equals("✕ 刪除")) {
    try {
        int iListIndex = jList1.getSelectedIndex();
        if(iListIndex >= 0) { //有選擇 List Item 才要刪除
            listModel1.removeElementAt(iListIndex);
            //新增 Vector 儲存左編 jList1,jList2,jList3 的順序

```

```
        for(int i=0;i < listModel1.size();i++) {
            }
            //end of 新增 Vector
        }
    }
    catch(Exception ex) {
    }
}
else if(e.getActionCommand().equals(">>")) {
    try {
        String strKey = String.valueOf(listModel1.toString());
        listModel2.addElement(strKey);
    }
    catch(Exception ex) {
    }
}
else{
    System.exit(0);
}
}
}
```



二、相關知識內容分析

```
package nkfust;
import java.io.*;
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import com.hp.hpl.jena.util.FileUtils;
import com.hp.hpl.jena.util.*;
import com.hp.hpl.jena.ontology.OntModel;
import edu.stanford.smi.protege.owl.*;
import edu.stanford.smi.protege.owl.jena.*;
import edu.stanford.smi.protege.owl.model.*;
import edu.stanford.smi.protege.owl.model.event.*;
import edu.stanford.smi.protege.owl.ui.*;
import edu.stanford.smi.protege.owl.ui.icons.*;
import edu.stanford.smi.protege.owl.ui.widget.*;
public class OWLProperty extends JFrame {
    public JenaOWLModel owlModelPublic;
    public static String strGetInsName;
    OWLNamedClass destinationClass;
    Vector vctInstanceList = new Vector();
    Vector vctTemp = new Vector();
    Hashtable htbProperty = new Hashtable();
    Vector vctReturn = new Vector();
    public OWLProperty() {
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                dispose();
                System.exit(0);
            }
        });
    }
};
```

```

    }
    public static void main(String[] args) throws Exception {
    }

    public Vector getProperty(String ClassName) throws Exception {
        String strSourceLocation = "C://nkfust//EIS.owl";
        FileInputStream fis;
        fis = new FileInputStream(new File(strSourceLocation));
        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        OWLNamedClass destinationClass =
        owlModelPublic.getOWLNamedClass(ClassName);
        ListPanel listPanel = new ListPanel(destinationClass);
        return vctReturn;
    }
    public Vector getPropertyFromIndividual(String IndividualName) throws
Exception {
        String strSourceLocation = "C://nkfust//EIS.owl";
        FileInputStream fis;
        Vector vtcTemp1 = new Vector();
        Vector vtcTemp2 = new Vector();
        Vector vtcTemp3 = new Vector();
        fis = new FileInputStream(new File(strSourceLocation));
        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        System.out.println("getPropertyFromIndividual:Start"+IndividualName);
        //從某一"事件"的"適用於"屬性抓 individual
        OWLIndividual userInstance =
        owlModelPublic.getOWLIndividual(IndividualName);
        System.out.println(userInstance.getBrowserText());
        RDFProperty NewProperty = owlModelPublic.getRDFProperty("產品名稱");
        System.out.println(NewProperty.getBrowserText());
        System.out.println(userInstance.getPropertyValueCount(NewProperty));
        OWLNamedClass targetClass;
        RDFProperty tmpProperty = owlModelPublic.getRDFProperty("負責人");
        //individual 可為多個，針對每一個去抓"使用結構" 屬性中的 Class

```

```

for (Iterator itt = userInstance.listPropertyValues(NewProperty); itt.hasNext();) {
    OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
for (Iterator ip = tmpIndividual.listPropertyValues(tmpProperty); ip.hasNext();) {
    targetClass = (OWLNamedClass) ip.next();
    System.out.println(targetClass.getBrowserText());
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(targetClass.getBrowserText());
    ListPanel listPanel = new ListPanel(destinationClass);
    }
}
return vctReturn;
}

//區域-庫存據點,銷售據點
public Vector getExampleFromClass1(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("負責人");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("姓名");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("月薪");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("年份");
    RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("月份");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        String strIndividual = individual.getBrowserText();
        System.out.println("Individual:"+strIndividual);
    }
}

```

```

for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
    OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
    System.out.println("Instance:"+tmpIndividual.getBrowserText());
    /individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("姓名："+strLawLevelProperty+" ");
vctTemp2.addElement("月薪："+strLawLevelProperty1+" ");
vctTemp3.addElement("年份："+strLawLevelProperty2+" ");
vctTemp4.addElement("月份："+strLawLevelProperty3+" ");
}
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
vctReturn.addElement(vctTemp4);
return vctReturn;
}

```

//區域-製造廠房

```

public Vector getExampleFromClass2(String ClassName) throws Exception {
String strSourceLocation = "C://nkfust//EIS.owl";
FileInputStream fis;
Vector vtcTemp1 = new Vector();
Vector vtcTemp2 = new Vector();
Vector vtcTemp3 = new Vector();
Vector vtcTemp4 = new Vector();

```

```

        Vector vtcTemp5 = new Vector();
        fis= new FileInputStream(new File(strSourceLocation));
        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
        RDFProperty NewProperty = owlModelPublic.getRDFProperty("貨品");
        RDFProperty tmpProperty = owlModelPublic.getRDFProperty("生產成本");
        RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("生產數量");
        RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("生產名稱");
        RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("年份");
        RDFProperty tmpProperty4 = owlModelPublic.getRDFProperty("月份");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
OWLIndividual individual = (OWLIndividual) it.next();
String strIndividual = individual.getBrowserText();
System.out.println("Individual:"+strIndividual);
for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
Object objLawLevelProperty4 = tmpIndividual.getPropertyValue(tmpProperty4);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
String strLawLevelProperty4 = String.valueOf(objLawLevelProperty4);
        System.out.println("Slot:"+strLawLevelProperty);
        vtcTemp1.addElement("生產成本："+strLawLevelProperty+" ");
        vtcTemp2.addElement("生產數量："+strLawLevelProperty1+" ");
        vtcTemp3.addElement("生產名稱："+strLawLevelProperty2+" ");
        vtcTemp4.addElement("年份："+strLawLevelProperty3+" ");

```



```

        vtcTemp5.addElement("月份 : "+strLawLevelProperty4+" ");
    }
}
vctReturn.addElement(vtcTemp1);
vctReturn.addElement(vtcTemp2);
vctReturn.addElement(vtcTemp3);
vctReturn.addElement(vtcTemp4);
vctReturn.addElement(vtcTemp5);
return vctReturn;
}

//區域-客戶資料
public Vector getExampleFromClass3(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    Vector vtcTemp5 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("銷售產品");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("產品名稱");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("數量");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("價格");
    RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("年份");
    RDFProperty tmpProperty4 = owlModelPublic.getRDFProperty("月份");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        String strIndividual = individual.getBrowserText();
        System.out.println("Individual:"+strIndividual);
    }
    for (Iterator itt = individual.listPropertyValues(NewProperty);

```

```

itt.hasNext();) {
    OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
    System.out.println("Instance:"+tmpIndividual.getBrowserText());
    //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
    Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
    Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
    Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
    Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
    Object objLawLevelProperty4 = tmpIndividual.getPropertyValue(tmpProperty4);
    String strLawLevelProperty = String.valueOf(objLawLevelProperty);
    String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
    String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
    String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
    String strLawLevelProperty4 = String.valueOf(objLawLevelProperty4);
    System.out.println("Slot:"+strLawLevelProperty);
    vtcTemp1.addElement("產品名稱："+strLawLevelProperty+"");
    vtcTemp2.addElement("數量："+strLawLevelProperty1+"");
    vtcTemp3.addElement("價格："+strLawLevelProperty2+"");
    vtcTemp4.addElement("年份："+strLawLevelProperty3+"");
    vtcTemp5.addElement("月份："+strLawLevelProperty4+"");
    }
}
vctReturn.addElement(vtcTemp1);
vctReturn.addElement(vtcTemp2);
vctReturn.addElement(vtcTemp3);
vctReturn.addElement(vtcTemp4);
vctReturn.addElement(vtcTemp5);
return vctReturn;
}

```

//員工-行銷部門

```

public Vector getExampleFromClass4(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;

```

```

    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    fis = new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(className);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("負責銷售點");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("地區");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("貨品");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("負責人");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
    for (Iterator itt = individual.listPropertyValues(NewProperty);
itt.hasNext();)
    {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
        Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
        Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
        Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
        String strLawLevelProperty = String.valueOf(objLawLevelProperty);
        String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
        String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
        System.out.println("Slot:"+strLawLevelProperty);
        vtcTemp1.addElement("地區 : "+strLawLevelProperty+" ");
        vtcTemp2.addElement("貨品 : "+strLawLevelProperty1+" ");
        vtcTemp3.addElement("負責人 : "+strLawLevelProperty2+" ");
    }
}
vctReturn.addElement(vtcTemp1);

```

```

        vctReturn.addElement(vtcTemp2);
        vctReturn.addElement(vtcTemp3);
        return vctReturn;
    }

    //員工-生產部門
    public Vector getExampleFromClass8(String ClassName) throws Exception {
        String strSourceLocation = "C://nkfust//EIS.owl";
        FileInputStream fis;
        Vector vtcTemp1 = new Vector();
        Vector vtcTemp2 = new Vector();
        Vector vtcTemp3 = new Vector();
        fis = new FileInputStream(new File(strSourceLocation));
        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        OWLNamedClass destinationClass =
        owlModelPublic.getOWLNamedClass(ClassName);
        RDFProperty NewProperty = owlModelPublic.getRDFProperty("負責廠房");
        RDFProperty tmpProperty = owlModelPublic.getRDFProperty("廠房名稱");
        RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("負責人");
        RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("地區");
        for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
            OWLIndividual individual = (OWLIndividual) it.next();
            String strIndividual = individual.getBrowserText();
            System.out.println("Individual:"+strIndividual);
            for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
                OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
                System.out.println("Instance:"+tmpIndividual.getBrowserText());
                //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
                Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
                Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
                Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
                String strLawLevelProperty = String.valueOf(objLawLevelProperty);
                String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
                String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
            }
        }
    }

```

```

System.out.println("Slot:"+strLawLevelProperty);
    vtcTemp1.addElement("廠房名稱："+strLawLevelProperty+"");
    vtcTemp2.addElement("負責人："+strLawLevelProperty1+"");
    vtcTemp3.addElement("地區："+strLawLevelProperty2+"");
        }
    }
    vctReturn.addElement(vtcTemp1);
    vctReturn.addElement(vtcTemp2);
    vctReturn.addElement(vtcTemp3);
    return vctReturn;
}

//員工-資訊部門
public Vector getExampleFromClass9(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    fis = new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("負責機器");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("機器名稱");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("月份");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("年份");
    RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("機器設備成本");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        String strIndividual = individual.getBrowserText();
        System.out.println("Individual:"+strIndividual);
    }
    for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {

```

```

        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("機器名稱："+strLawLevelProperty+" ");
vctTemp2.addElement("月份："+strLawLevelProperty1+" ");
vctTemp3.addElement("年份："+strLawLevelProperty2+" ");
vctTemp4.addElement("機器設備成本："+strLawLevelProperty3+" ");
    }
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
vctReturn.addElement(vctTemp4);
return vctReturn;
}
//運輸
public Vector getExampleFromClass5(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    Vector vtcTemp5 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));

```

```

        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
RDFProperty NewProperty = owlModelPublic.getRDFProperty("客戶");
RDFProperty tmpProperty = owlModelPublic.getRDFProperty("公司名稱");
RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("信用評等");
RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("購買數量");
RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("年份");
RDFProperty tmpProperty4 = owlModelPublic.getRDFProperty("月份");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
OWLIndividual individual = (OWLIndividual) it.next();
String strIndividual = individual.getBrowserText();
System.out.println("Individual:"+strIndividual);
for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
Object objLawLevelProperty4 = tmpIndividual.getPropertyValue(tmpProperty4);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
String strLawLevelProperty4 = String.valueOf(objLawLevelProperty4);
System.out.println("Slot:"+strLawLevelProperty);
vtcTemp1.addElement("公司名稱："+strLawLevelProperty+" ");
vtcTemp2.addElement("信用評等："+strLawLevelProperty1+" ");
vtcTemp3.addElement("購買數量："+strLawLevelProperty2+" ");
vtcTemp4.addElement("年份："+strLawLevelProperty3+" ");
vtcTemp5.addElement("月份："+strLawLevelProperty4+" ");
}
}

```

```

    }
    vctReturn.addElement(vtcTemp1);
    vctReturn.addElement(vtcTemp2);
    vctReturn.addElement(vtcTemp3);
    vctReturn.addElement(vtcTemp4);
    vctReturn.addElement(vtcTemp5);
    return vctReturn;
}

// 銷售量
public Vector getExampleFromClass6(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    Vector vtcTemp5 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("銷售產品");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("產品名稱");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("銷售數量");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("銷售金額");
    RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("月份");
    RDFProperty tmpProperty4 = owlModelPublic.getRDFProperty("年份");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        String strIndividual = individual.getBrowserText();
        System.out.println("Individual:"+strIndividual);
    }
    for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();

```



```

        System.out.println("Instance:"+tmpIndividual.getBrowerText());
//individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
Object objLawLevelProperty4 = tmpIndividual.getPropertyValue(tmpProperty4);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
String strLawLevelProperty4 = String.valueOf(objLawLevelProperty4);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("產品名稱："+strLawLevelProperty+" ");
vctTemp2.addElement("銷售數量："+strLawLevelProperty1+" ");
vctTemp3.addElement("銷售金額："+strLawLevelProperty2+" ");
vctTemp4.addElement("月份："+strLawLevelProperty3+" ");
vctTemp5.addElement("年份："+strLawLevelProperty4+" ");
    }
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
vctReturn.addElement(vctTemp4);
vctReturn.addElement(vctTemp5);
return vctReturn;
}

//生產
public Vector getExampleFromClass7(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vctTemp1 = new Vector();
    Vector vctTemp2 = new Vector();

```

```

Vector vtcTemp3 = new Vector();
Vector vtcTemp4 = new Vector();
Vector vtcTemp5 = new Vector();
fis= new FileInputStream(new File(strSourceLocation));
owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(className);
RDFProperty NewProperty = owlModelPublic.getRDFProperty("所需原物料");
RDFProperty tmpProperty = owlModelPublic.getRDFProperty("原物料名稱");
RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("原物料成本");
RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("原物料價格");
RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("月份");
RDFProperty tmpProperty4 = owlModelPublic.getRDFProperty("年份");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
    OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
    System.out.println("Instance:"+tmpIndividual.getBrowserText());
//individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
Object objLawLevelProperty4 = tmpIndividual.getPropertyValue(tmpProperty4);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
String strLawLevelProperty4 = String.valueOf(objLawLevelProperty4);
System.out.println("Slot:"+strLawLevelProperty);
vtcTemp1.addElement("原物料名稱："+strLawLevelProperty+"");
vtcTemp2.addElement("原物料成本："+strLawLevelProperty1+"");

```

```

vctTemp3.addElement("原物料價格："+strLawLevelProperty2+"");
vctTemp4.addElement("月份："+strLawLevelProperty3+"");
vctTemp5.addElement("年份："+strLawLevelProperty4+"");
    }
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
vctReturn.addElement(vctTemp4);
vctReturn.addElement(vctTemp5);
return vctReturn;
}

//機器設備
public Vector getExampleFromClass10(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";

    FileInputStream fis;
    Vector vctTemp1 = new Vector();
    Vector vctTemp2 = new Vector();
    Vector vctTemp3 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
    owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("維修人員");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("姓名");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("月薪");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("負責機器");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
    for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {

```

```

OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
System.out.println("Instance:"+tmpIndividual.getBrowserText());

//individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("姓名："+strLawLevelProperty+"");
vctTemp2.addElement("月薪："+strLawLevelProperty1+"");
vctTemp3.addElement("負責機器："+strLawLevelProperty2+"");
    }
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
return vctReturn;
}

//庫存
public Vector getExampleFromClass11(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    fis= new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
    owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("盤點人");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("姓名");

```

```

RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("職位");
RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("月薪");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
    for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());

        //individual 可為多個，針對每一個去抓"條文內容" 屬性中的字串
        Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
        Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
        Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
        String strLawLevelProperty = String.valueOf(objLawLevelProperty);
        String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
        String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
        System.out.println("Slot:"+strLawLevelProperty);
        vtcTemp1.addElement("姓名："+strLawLevelProperty+"");
        vtcTemp2.addElement("職位："+strLawLevelProperty1+"");
        vtcTemp3.addElement("月薪："+strLawLevelProperty2+"");
    }
}
vctReturn.addElement(vtcTemp1);
vctReturn.addElement(vtcTemp2);
vctReturn.addElement(vtcTemp3);
return vctReturn;
}

//原物料
public Vector getExampleFromClass13(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust/EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();

```

```

        Vector vtcTemp2 = new Vector();
        Vector vtcTemp3 = new Vector();
        Vector vtcTemp4 = new Vector();
        fis = new FileInputStream(new File(strSourceLocation));
        owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
        OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
RDFProperty NewProperty = owlModelPublic.getRDFProperty("原料加工成品");
RDFProperty tmpProperty = owlModelPublic.getRDFProperty("生產名稱");
RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("生產數量");
RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("月份");
RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("年份");
for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
    for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
        Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
        Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
        Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
        String strLawLevelProperty = String.valueOf(objLawLevelProperty);
        String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
        String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
        String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
        System.out.println("Slot:"+strLawLevelProperty);
        vtcTemp1.addElement("生產名稱："+strLawLevelProperty+"");
        vtcTemp2.addElement("生產數量："+strLawLevelProperty1+"");
        vtcTemp3.addElement("月份："+strLawLevelProperty2+"");
        vtcTemp4.addElement("年份："+strLawLevelProperty3+"");
    }
}

```

```

        vctReturn.addElement(vtcTemp1);
        vctReturn.addElement(vtcTemp2);
        vctReturn.addElement(vtcTemp3);
        vctReturn.addElement(vtcTemp4);
        return vctReturn;
    }

```

//產品

```

public Vector getExampleFromClass14(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    Vector vtcTemp4 = new Vector();
    fis = new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("銷售人員");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("負責商品");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("負責銷售點");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("姓名");
    RDFProperty tmpProperty3 = owlModelPublic.getRDFProperty("職位");
    for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        String strIndividual = individual.getBrowserText();
        System.out.println("Individual:"+strIndividual);
        for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
            OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
            System.out.println("Instance:"+tmpIndividual.getBrowserText());
            Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
            Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
            Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);

```

```

Object objLawLevelProperty3 = tmpIndividual.getPropertyValue(tmpProperty3);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
String strLawLevelProperty3 = String.valueOf(objLawLevelProperty3);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("負責商品："+strLawLevelProperty+"");
vctTemp2.addElement("負責銷售點："+strLawLevelProperty1+"");
vctTemp3.addElement("姓名："+strLawLevelProperty2+"");
vctTemp4.addElement("職位："+strLawLevelProperty3+"");
    }
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
vctReturn.addElement(vctTemp4);
return vctReturn;
}

//市場狀況
public Vector getExampleFromClass15(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    fis = new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
    OWLNamedClass destinationClass =
    owlModelPublic.getOWLNamedClass(ClassName);
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("公共工程");
    RDFProperty tmpProperty = owlModelPublic.getRDFProperty("公共工程價格");
    RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("公共工程名稱");
    RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("公共工程數量");

```



```

for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
    OWLIndividual individual = (OWLIndividual) it.next();
    String strIndividual = individual.getBrowserText();
    System.out.println("Individual:"+strIndividual);
    for (Iterator itt = individual.listPropertyValue(NewProperty); itt.hasNext();) {
        OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
        System.out.println("Instance:"+tmpIndividual.getBrowserText());
        Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
        Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
        Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
        String strLawLevelProperty = String.valueOf(objLawLevelProperty);
        String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
        String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
        System.out.println("Slot:"+strLawLevelProperty);
        vtcTemp1.addElement("公共工程價格："+strLawLevelProperty+" ");
        vtcTemp2.addElement("公共工程名稱："+strLawLevelProperty1+" ");
        vtcTemp3.addElement("公共工程數量："+strLawLevelProperty2+" ");
    }
}
vctReturn.addElement(vtcTemp1);
vctReturn.addElement(vtcTemp2);
vctReturn.addElement(vtcTemp3);
return vctReturn;
}

//競爭者
public Vector getExampleFromClass16(String ClassName) throws Exception {
    String strSourceLocation = "C://nkfust//EIS.owl";
    FileInputStream fis;
    Vector vtcTemp1 = new Vector();
    Vector vtcTemp2 = new Vector();
    Vector vtcTemp3 = new Vector();
    fis = new FileInputStream(new File(strSourceLocation));
    owlModelPublic = ProtegeOWL.createJenaOWLModelFromInputStream(fis);
}

```

```

        OWLNamedClass destinationClass =
owlModelPublic.getOWLNamedClass(ClassName);
RDFProperty NewProperty = owlModelPublic.getRDFProperty("比較市場狀況");
RDFProperty tmpProperty = owlModelPublic.getRDFProperty("產品名稱");
RDFProperty tmpProperty1 = owlModelPublic.getRDFProperty("數量");
RDFProperty tmpProperty2 = owlModelPublic.getRDFProperty("價格");

for (Iterator it = destinationClass.getInstances(true).iterator(); it.hasNext();) {
OWLIndividual individual = (OWLIndividual) it.next();
String strIndividual = individual.getBrowserText();
System.out.println("Individual:"+strIndividual);
for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
OWLIndividual tmpIndividual = (OWLIndividual) itt.next();
System.out.println("Instance:"+tmpIndividual.getBrowserText());
Object objLawLevelProperty = tmpIndividual.getPropertyValue(tmpProperty);
Object objLawLevelProperty1 = tmpIndividual.getPropertyValue(tmpProperty1);
Object objLawLevelProperty2 = tmpIndividual.getPropertyValue(tmpProperty2);
String strLawLevelProperty = String.valueOf(objLawLevelProperty);
String strLawLevelProperty1 = String.valueOf(objLawLevelProperty1);
String strLawLevelProperty2 = String.valueOf(objLawLevelProperty2);
System.out.println("Slot:"+strLawLevelProperty);
vctTemp1.addElement("產品名稱："+strLawLevelProperty+"");
vctTemp2.addElement("數量："+strLawLevelProperty1+"");
vctTemp3.addElement("價格："+strLawLevelProperty2+"");
}
}
vctReturn.addElement(vctTemp1);
vctReturn.addElement(vctTemp2);
vctReturn.addElement(vctTemp3);
return vctReturn;
}
private class ListPanel extends JPanel {
private OWLNamedClass destinationClass;
private JList list;

```

```

private DefaultListModel listModel;
private OWLModel owlModel;
private ModelListener modelListener = new ModelAdapter() {
    public void individualCreated(RDFResource resource) {
        if (resource.hasRDFType(destinationClass, true)) {
            handleDestinationAdded(resource);
        }
    }
};
ListPanel(OWLNamedClass activityClass) {
    this.destinationClass = activityClass;
    this.owlModel = activityClass.getOWLModel();
    owlModel.addModelListener(modelListener);
    listModel = new DefaultListModel();
    String strIndividual = "";
    RDFProperty NewProperty = owlModelPublic.getRDFProperty("職位");
    for (Iterator it = activityClass.getInstances(true).iterator(); it.hasNext();) {
        OWLIndividual individual = (OWLIndividual) it.next();
        strIndividual = individual.getBrowserText();
        if (!htbProperty.containsKey(strIndividual)) {
            vctInstanceList.add(strIndividual);
            System.out.println("Individual:"+strIndividual);
            for (Iterator itt = individual.listPropertyValues(NewProperty); itt.hasNext();) {
                String proper = String.valueOf(itt.next());
                System.out.println("Property:"+proper);
                vctTemp.add(proper);
            }
            htbProperty.put(strIndividual, vctTemp);
        }
    }
    vctReturn.add(vctInstanceList);
    vctReturn.add(htbProperty);
    OWLLabeledComponent lc = new OWLLabeledComponent("格式列表", new
    JScrollPane(list));

```

```

        setLayout(new BorderLayout());
        add(BorderLayout.CENTER, lc);
    }
    private void addDestination() {
        OWLNamedClass newType =
OWLUI.pickOWLNamedClass(owlModel,
Collections.singleton(destinationClass), "Select type of new Destination");
        if (newType != null) {
            String name = JOptionPane.showInputDialog("Enter name of
new " + newType.getBrowserText());
                if (name != null) {
                    newType.createOWLIndividual(name);
                }
            }
        }
    public void dispose() {
        owlModel.removeModelListener(modelListener);
    }
    private void handleDestinationAdded(RDFResource destination) {
        listModel.addElement(destination);
        list.setSelectedValue(destination, true);
    }
}
}
}

```